EACL 2014

14th Conference of the European Chapter of the Association for Computational Linguistics

Proceedings of the Conference (Volume 2: Short Papers)

April 26-30, 2014
Gothenburg, Sweden

GOLD SPONSORS

SILVER SPONSOR

BRONZE SPONSORS

SUPPORTERS

EXHIBITORS

OTHER SPONSORS

HOSTS

CLT
Centre for Language Technology
Gothenburg, Sweden

Order copies of this and other ACL proceedings from:

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

ISBN 978-1-937284-99-2

Preface: General Chair

Welcome to EACL 2014, the 14th Conference of the European Chapter of the Association for Computational Linguistics! This is the largest EACL meeting ever: with eighty long papers, almost fifty short ones, thirteen student research papers, twenty-six demos, fourteen workshops and six tutorials, we expect to bring to Gothenburg up to five hundred participants, for a week of excellent science interspersed with entertaining social events.

It is hard to imagine how much work is involved in the preparation of such an event. It takes about three years, from the day the EACL board starts discussing the location and nominating the chairs, until the final details of the budget are resolved. The number of people involved is also huge, and I was fortunate to work with an excellent, dedicated and efficient team, to which I am enormously grateful.

The scientific program was very ably composed by the Program Committee Chairs, Sharon Goldwater and Stefan Riezler, presiding over a team of twenty-four area chairs. Given that this year we had long paper submissions, followed by a rebuttal period, followed by a very stressed short paper reviewing period, this meant *a lot* of work. Overall, Sharon and Stefan handled over five hundred submissions, or over 1,500 reviews! The result of this work is a balanced, high-quality scientific program that I'm sure we will all enjoy. The PC Chairs have also selected the three invited speakers, and we will have the pleasure of attending keynotes delivered by Simon King, Ulrike von Luxburg, and Dan Roth – a great choice of speakers!

The diverse workshop program was put together by the Workshop Chairs, Anja Belz and Reut Tsarfaty, under very strict deadlines due to the fact that as in previous years, workshops were coordinated with other ACL events (this year, ACL and EMNLP). Even in light of the competition, Anja and Reut negotiated a varied and attractive set of fourteen workshops which will keep us busy over the weekend prior to the main conference.

Also on that weekend are the six tutorials, selected from among several submissions by the Tutorial Chairs, Afra Alishahi and Marco Baroni. Again, the tutorials offer a set of diverse and timely topics, covering both core areas of NLP and tangential fields of research.

We included in the program a large number of demonstrations, selected by Marko Tadić and Bogdan Babych, the Demo Chairs. And an integral part of the scientific program is the Student Research Workshop, put together by the SRW Chairs, Desmond Elliott, Konstantina Garoufi, Douwe Kiela, and Ivan Vulić, whose work was supervised by the SRW Faculty Advisor, Sebastian Padó.

The Proceedings that you're reading now were compiled by the Publication Chairs, Gosse Bouma and Yannick Parmentier. Their responsibilities include the preparation of all the proceedings, including the main session, the SRW, the demo session, the workshop proceedings etc. – thousands of pages, all under very strict deadlines.

It has been a very special pleasure for me to work with an excellent local organization team. The Local Organization Chairs, Lars Borin and Aarne Ranta, were assisted by an extremely efficient team, Yvonne Adesam, Martin Kaså and Nina Tahmasebi. Their effort cannot be overestimated: from dealing with the two universities over issues of conference space and funding, through dealing with two professional conference organizers, to corresponding with authors, participants and of course all the other chairs. Add the stress involved in being in charge of a hefty budget that has to be balanced by the end of the conference, and you can only admire the relaxed way in which they took upon themselves this daunting task.

The local team included also Peter Ljunglöf, the Publicity Chair, to whom we should all be grateful for the beautiful web site of the conference and the timely e-mails, tweets and Facebook statuses. The Local Sponsorship Chairs, Sofie Johansson Kokkinakis and Staffan Larsson, worked together with the ACL

Sponsorship Chairs Jochen Leidner and Alessandro Moschitti, to obtain some much needed financial support. Sincere thanks are due to the various sponsors for their generous contribution.

The local team did a wonderful job organizing a social program this year. This includes a reception at the City Hall on Sunday, a catered poster and demo session on Monday, a conference dinner on Tuesday and of course, the famous Cortège at the very end of the conference. A perfect mix of business and pleasure.

I am grateful to all members of the EACL board for their advice and guidance, and in particular to past Chair Sien Moens, Chair Stephen Clark, Chair-elect Lluìs Màrquez and Treasurer Mike Rosner. Many thanks are also due to the ACL Treasurer Graeme Hirst and of course, as always, to the ACL Business Manager Priscilla Rasmussen, who was always there with her vast experience to clear up uncertainties and lend a helping hand.

Finally, let us not forget that this is all about *you*: authors, reviewers, demo presenters, workshop organizers and speakers, tutorial speakers and participants of the conference. Thank you for choosing to be part of EACL-2014, I wish you a very enjoyable conference!

Shuly Wintner, University of Haifa
General Chair
March 2014

Preface: Program Chairs

We are delighted to present you with this volume containing the papers accepted for presentation at the 14th Conference of the European Chapter of the Association for Computational Linguistics, held in Gothenburg, Sweden, from April 26 till April 30 2014.

EACL 2014 introduced a short paper (4 page) format in addition to the usual long paper (8 page) format, which led to the highest total number of submissions of any EACL. We received 317 valid long paper submissions and were able to accept 78 of these papers (an acceptance rate of 24.6%). 49 of the papers (15.4%) were accepted for oral presentation, and 31 (9.8%) for poster presentation. In addition, we received 199 valid short paper submissions and were able to accept 46 of these (an acceptance rate of 23.1accepted for oral presentation, and 13 (6.5%) for poster presentation. The EACL 2014 schedule also includes oral presentations from two papers published in the Transactions of the Association for Computational Linguistics, a new feature of this year's conference.

The introduction of short papers, handled in a second round of submissions, meant a somewhat higher workload for our program committee, and we are very grateful to our 24 area chairs for recruiting an excellent panel of 434 reviewers from all over the world, and to those reviewers for providing their feedback on the submissions. Each submission was reviewed by at least three reviewers (at least two for short papers), who were then encouraged to discuss any differences of opinion, taking into account the responses of the authors to their initial reviews. Based on the reviews, author response, and reviewer discussion, area chairs provided a ranking for papers in their area. Final selection was made by the program co-chairs after discussion with the area chairs and an independent check of reviews.

Each area chair was also asked to nominate the best long paper and best short paper from his or her area, or to decline to nominate any. Several papers were nominated, and of these the program co-chairs made the final decision on the Best Long Paper and Best Short Paper awards, which will be awarded in a plenary session at the conference.

In addition to the main conference program, EACL 2014 will feature the now traditional Student Research Workshop, 14 other workshops, 6 tutorials and a demo session with 26 presentations. We are also fortunate to have three excellent invited speakers: Dan Roth (University of Illinois at Urbana-Champaign), Ulrike von Luxburg (University of Hamburg), and Simon King (University of Edinburgh).

We would very much like to thank all of the other people who have helped us put together this year's conference. Most importantly, all of the authors who submitted their work to EACL, without whom we would have no conference at all! The number and quality of both long and short paper submissions in many different areas shows that we are maintaining and growing a broad and active community. We are greatly indebted to all the area chairs and reviewers for their hard work, which allowed us to choose from amongst the many high-quality submissions to put together a strong programme and provide useful feedback to authors. The START support team, and especially Rich Gerber, were of great help in swiftly answering all of our technical questions, and occasionally even knowing more about our job than we did! We thank the invited speakers for agreeing to present at EACL, and the publication chairs, Yannick Parmentier and Gosse Bouma, for putting this volume together. The local organizing committee (Lars Borin, Aarne Ranta, Yvonne Adesam, Martin Kaså, and Nina Tahmasebi) have been invaluable in arranging the logistics of the conference and coordinating with us on many organizational issues, and we are grateful to the publicity chair, Peter Ljunglöf, for ensuring up-to-date programme information on the conference web site. We thank also the Student Research Workshop chairs for smoothly coordinating with us on their schedule. Last but not least, we are indebted to the General Chair, Shuly Wintner, for his guidance and support throughout the whole process.

We hope you enjoy the conference!

Sharon Goldwater and Stefan Riezler
EACL 2014 Programme Chairs

General Chair:

Shuly Wintner, University of Haifa (Israel)

Program Chairs:

Sharon Goldwater, University of Edinburgh (UK)
Stefan Riezler, Heidelberg University (Germany)

Local Organizing Committee:

Lars Borin (chair), University of Gothenburg (Sweden)
Aarne Ranta (chair), University of Gothenburg and Chalmers University of Technology (Sweden)
Yvonne Adesam, University of Gothenburg (Sweden)
Martin Kaså, University of Gothenburg (Sweden)
Nina Tahmasebi, Chalmers University of Technology (Sweden)

Publication Chairs:

Gosse Bouma, University of Groningen (The Netherlands)
Yannick Parmentier, University of Orléans (France)

Workshop Chairs:

Anja Belz, University of Brighton (UK)
Reut Tsarfaty, Uppsala University (Sweden)

Tutorial Chairs:

Afra Alishahi, Tilburg University (The Netherlands)
Marco Baroni, University of Trento (Italy)

Demo Chair:

Marko Tadić, University of Zagreb (Croatia)
Bogdan Babych, University of Leeds (UK)

Student Research Workshop Chairs:

Desmond Elliott, University of Edinburgh (UK)
Konstantina Garoufi, University of Potsdam (Germany)
Douwe Kiela, University of Cambridge (UK)
Ivan Vulić, KU Leuven (Belgium)

Student Research Workshop Faculty advisor:

Sebastian Padó, Heidelberg University (Germany)

Sponsorship Chairs:

Jochen Leidner, Thomson-Reuters/Linguit Ltd. (Switzerland)

Alessandro Moschitti, University of Trento (Italy)
Sofie Johansson Kokkinakis, University of Gothenburg (Sweden)
Staffan Larsson, University of Gothenburg (Sweden)

Publicity Chair:

Peter Ljunglöf, University of Gothenburg and Chalmers University of Technology (Sweden)

Area Chairs:

Enrique Alfonseca, John Blitzer, Aoife Cahill, Vera Demberg, Chris Dyer, Jacob Eisenstein, Micha Elsner, Katrin Erk, Afsaneh Fazly, Katja Filippova, Alexander Fraser, Iryna Gurevych, Chin-Yew Lin, David McClosky, Yusuke Miyao, Hwee Tou Ng, Slav Petrov, Simone Paolo Ponzetto, Sebastian Riedel, Verena Rieser, Helmut Schmid, Izhak Shafran, Hiroya Takamura, Lucy Vanderwende

Reviewers:

Fadi Abu-Sheika, Meni Adler, Nitish Aggarwal, Lars Ahrenberg, Afra Alishahi, Yaser Al-Onaizan, Yasemin Altun, Waleed Ammar, Jacob Andreas, Ion Androutsopoulos, Gabor Angeli, Mihael Arcan, Yoav Artzi, Jordi Atserias Batalla, Michael Auli, Harald Baayen, Timothy Baldwin, David Bamman, Mohit Bansal, Marco Baroni, Loïc Barrault, Núria Bel, Kedar Bellare, Islam Beltagy, Luciana Benotti, Yinon Bentor, Jonathan Berant, Sabine Bergler, Raffaella bernardi, Klinton Bicknell, Chris Biemann, Arianna Bisazza, Yonatan Bisk, Roi Blanco, Michael Bloodgood, Phil Blunsom, Nathan Bodenstab, Branimir Boguraev, Bernd Bohnet, Gemma Boleda, Danushka Bollegala, Francis Bond, Kalina Bontcheva, Johan Bos, Houda Bouamor, Thorsten Brants, Chloé Braud, Fabienne Braune, Chris Brew, Ted Briscoe, Julian Brooke, Marco Brunello, Paul Buitelaar, Harry Bunt, Aljoscha Burchardt, David Burkett, Stephan Busemann, Bill Byrne, Nicoletta Calzolari, Ivan Cantador, Yunbo Cao, Guiseppe Carenini, Marine Carpuat, Xavier Carreras, John Carroll, Dave Carter, Francisco Casacuberta, Pablo Castells, Nathanael Chambers, Jason Chang, Ming-Wei Chang, David Chen, Hsin-Hsi Chen, Wenliang Chen, Chen Chen, Kehai Chen, Colin Cherry, Jackie Chi Kit Cheung, David Chiang, Christian Chiarcos, Kostadin Cholakov, Christos Christodoulopoulos, Jennifer Chu-Carroll, Cindy Chung, Massimiliano Ciaramita, Philipp Cimiano, Stephen Clark, Shay Cohen, Bonaventura Coppola, Marta R. Costa-jussà, Danilo Croce, Heriberto Cuayahuitl, Walter Daelemans, Cristian Danescu-Niculescu-Mizil, Dipanjan Das, Brian Davis, Munmun De Choudhury, Marie-Catherine de Marneffe, Gerard de Melo, Thierry Declerck, Michael Deisher, Steve DeNeefe, John DeNero, Pascal Denis, Michael Denkowski, Leon Derczynski, Marilena di Bari, Barbara Di Eugenio, Alberto Diaz, Michelangelo Diligenti, Markus Dreyer, Gregory Druck, Jinhua Du, Xiangyu Duan, Kevin Duh, Ewan Dunbar, Nadir Durrani, Marc Dymetman, Judith Eckle-Kohler, Koji Eguchi, Vladimir Eidelman, Andreas Eisele, David Elson, Angela Fahrni, James Fan, Richárd Farkas, Manaal Faruqui, Miriam Fernandez, Raquel Fernandez, Oliver Ferschke, João Filgueiras, Mark Fishel, Jeffrey Flanigan, Radu Florian, Mikel Forcada, Karën Fort, Eric Fosler-Lussier, Victoria Fossum, Jennifer Foster, Gil Francopoulo, Stefan L. Frank, Stella Frank, Francesca Frontini, Alona Fyshe, Michel Galley, Juri Ganitkevitch, Wenxuan Gao, Claire Gardent, Dan Garrette, Guillermo Garrido, Albert Gatt, Georgi Georgiev, Andrea Gesmundo, Arnab Ghoshal, George Giannakopoulos, Daniel Gildea, Kevin Gimpel, Jonathan Ginzburg, Yoav Goldberg, Julio Gonzalo, Spence Green, Edward Grefenstette, Camille Guinaudeau, Sonal Gupta, Francisco Guzman, Nizar Habash, Barry Haddow, John Hale, David Hall, Keith Hall, Greg Hanneman, Sanda Harabagiu, Christian Hardmeier, Matthias Hartung, mohammed hasanuzzaman, Katsuhiko Hayashi, Zhongjun He, Michael Heilman, James Henderson, John Henderson, Aurélie Herbelot, Ulf Hermjakob, Raquel Hervas, Graeme Hirst, Hieu Hoang, Johannes Hoffart, Mark Hopkins, Veronique Hoste, Fei Huang, Xiaojiang Huang, Xuanjing Huang, Rebecca Hwa, Nancy Ide, Gonzalo Iglesias, Diana Inkpen, Ann Irvine, Jagadeesh Jagarlamudi, Srinivasan Janarthanam, Lifeng Jia, Richard Johansson, Doug Jones, Laura Kallmeyer, Jaap Kamps, Evange-

los Kanoulas, Damianos Karakos, Graham Katz, Simon Keizer, Frank Keller, Shahram Khadivi, Adam Kilgarriff, Jin-Dong Kim, Seungyeon Kim, Katrin Kirchhoff, Philipp Koehn, Alexander Koller, Terry Koo, Anna Korhonen, Zornitsa Kozareva, Emiel Krahmer, Marco Kuhlmann, Roland Kuhn, Shankar Kumar, Jonathan Kummerfeld, Patrik Lambert, Phillippe Langlais, Guy Lapalme, Egoitz Laparra, Mirella Lapata, Staffan Larsson, Thomas Lavergne, Alon Lavie, Florian Laws, Lillian Lee, Junhui Li, lishuang li, Zhenghua Li, Maria Liakata, Chu-Cheng Lin, Krister Linden, Xiao Ling, Bing Liu, Jing Liu, Qun Liu, Yang Liu, Karen Livescu, Peter Ljunglöf, Elena Lloret, Adam Lopez, Annie Louis, Wei Lu, Yanjun Ma, Ji Ma, Klaus Macherey, Wolfgang Macherey, Bernardo Magnini, Inderjeet Mani, Chris Manning, Daniel Marcu, José B. Mariño, André F. T. Martins, Yuval Marton, Rebecca Mason, Yuji Matsumoto, Takuya Matsuzaki, Cettolo Mauro, Arne Mauser, Chandler May, Diana McCarthy, Ryan McDonald, Bob McMurray, Yashar Mehdad, Edgar Meij, Arul Menezes, Florian Metze, Christian M. Meyer, Jeffrey Micher, Bonan Min, Margaret Mitchell, Behrang Mohit, Karo Moilanen, Monica Monachini, Christof Monz, Raymond Mooney, Andrea Moro, Alessandro Moschitti, Thomas Mueller, Smaranda Muresan, Brian Murphy, Seung-Hoon Na, Tetsuji Nakagawa, Toshiaki Nakazawa, Preslav Nakov, Ramesh Nallapati, Vivi Nastase, Tetsuya Nasukawa, Roberto Navigli, Mark-Jan Nederhof, Sapna Negi, Matteo Negri, Ani Nenkova, Graham Neubig, Vincent Ng, Jian-Yun Nie, Jan Niehues, Joakim Nivre, Brendan O'Connor, Stephan Oepen, Kemal Oflazer, Naoaki Okazaki, Gozde Ozbal, Sebastian Padó, Martha Palmer, Patrick Pantel, Cecile Paris, Christopher Parisien, Rebecca J. Passonneau, Alexandre Passos, Siddharth Patwardhan, Michael Paul, Michael J. Paul, Adam Pauls, Sasa Petrovic, Daniele Pighin, Andrei Popescu-Belis, Maja Popović, Fred Popowich, Matt Post, Sameer Pradhan, John Prager, Stephen Pulman, Matthew Purver, Sampo Pyysalo, Behrang Qasemizadeh, Ariadna Quattoni, Chris Quirk, Altaf Rahman, Owen Rambow, Ari Rappoport, Sujith Ravi, Alexis Raykhel, Michaela Regneri, Roi Reichart, Ehud Reiter, Jason Riesa, German Rigau, Alan Ritter, Stephen Roller, Laurent Romary, Carolyn Rose, Michael Roth, Dana Rubinstein, Rachel Rudinger, Vasile Rus, Alexander M. Rush, Graham Russell, Delia Rusu, Kenji Sagae, Horacio Saggion, Kazi Saidul Hasan, Hassan Sajjad, Mark Sammons, Baskaran Sankaran, Felix Sasaki, Giorgio Satta, Hassan Sawaf, David Schlangen, Nathan Schneider, Björn Schuller, Sabine Schulte im Walde, Yohei Seki, Hendra Setiawan, Aliaksei Severyn, Serge Sharoff, Libin Shen, Shuming Shi, Hiroyuki Shindo, Ekaterina Shutova, Advaith Siddharthan, Carina Silberer, Mario J. Silva, Khalil Sima'an, Michel Simard, Kiril Simov, Serra Sinem Tekiroglu, Sameer Singh, Olivier Siohan, Gabriel Skantze, Nathaniel Smith, Stephen Soderland, Anders Søgaard, Thamar Solorio, Hagen Soltau, Swapna Somasundaran, Lucia Specia, Valentin Spitkovsky, Caroline Sporleder, Edward Stabler, Mark Steedman, Josef Steinberger, Georg Stemmer, Amanda Stent, Mark Stevenson, Matthew Stone, Veselin Stoyanov, Carlo Strapparava, Michael Strube, Sara Stymne, Keh-Yih Su, Katsuhito Sudoh, Weiwei Sun, Mihai Surdeanu, Jun Suzuki, Mary Swift, Stan Szpakowicz, Whitney Tabor, Partha Pratim Talukdar, Joel Tetreault, Simone Teufel, Stefan Thater, Mariët Theune, Blaise Thomson, Jörg Tiedemann, Christoph Tillmann, Kristina Toutanova, David Traum, Ming-Feng Tsai, Richard Tzong-Han Tsai, Ioannis Tsochantaridis, Yulia Tsvetkov, Dan Tufiş, Masao Utiyama, Tim Van de Cruys, Antal van den Bosch, Benjamin Van Durme, Josef van Genabith, Paola Velardi, David Vilar, Andreas Vlachos, Stephan Vogel, Clare Voss, Stephen Wan, Haifeng Wang, Kai Wang, ling wang, Wen Wang, Pidong Wang, Yu-Chun Wang, Taro Watanabe, Bonnie Webber, Jason Williams, Philip Williams, Colin Wilson, Travis Wolfe, Dekai Wu, Sander Wubben, Fei Xia, Deyi Xiong, Deyi Xiong, Peng Xu, Bishan Yang, Hui Yang, Muyun Yang, tae yano, Limin Yao, Dani Yogatama, François Yvon, Beñat Zapirain, Richard Zens, Torsten Zesch, Luke Zettlemoyer, Feifei Zhai, Hui Zhang, Joy Ying Zhang, Lei Zhang, Min Zhang, Yi Zhang, Yue Zhang, Meng Zhang, Liu Zhanyi, Shiqi Zhao, Tiejun Zhao, Xin Zhao, Xin Zhao, Muhua Zhu, Chengqing Zong

Table of Contents

Easy Web Search Results Clustering: When Baselines Can Reach State-of-the-Art Algorithms
Jose G. Moreno and Gaël Dias 1

Propagation Strategies for Building Temporal Ontologies
Mohammed Hasanuzzaman, Gaël Dias, Stéphane Ferrari and Yann Mathet 6

Chinese Open Relation Extraction for Knowledge Acquisition
Yuen-Hsien Tseng, Lung-Hao Lee, Shu-Yen Lin, Bo-Shun Liao, Mei-Jun Liu, Hsin-Hsi Chen, Oren Etzioni and Anthony Fader 12

Temporal Text Ranking and Automatic Dating of Texts
Vlad Niculae, Marcos Zampieri, Liviu Dinu and Alina Maria Ciobanu 17

Measuring the Similarity between Automatically Generated Topics
Nikolaos Aletras and Mark Stevenson 22

Projecting the Knowledge Graph to Syntactic Parsing
Andrea Gesmundo and Keith Hall 28

A Vague Sense Classifier for Detecting Vague Definitions in Ontologies
Panos Alexopoulos and John Pavlopoulos 33

Chasing Hypernyms in Vector Spaces with Entropy
Enrico Santus, Alessandro Lenci, Qin Lu and Sabine Schulte im Walde 38

Tight Integration of Speech Disfluency Removal into SMT
Eunah Cho, Jan Niehues and Alex Waibel 43

Non-Monotonic Parsing of Fluent Umm I mean Disfluent Sentences
Mohammad Sadegh Rasooli and Joel Tetreault 48

Lightly-Supervised Word Sense Translation Error Detection for an Interactive Conversational Spoken Language Translation System
Dennis Mehay, Sankaranarayanan Ananthakrishnan and Sanjika Hewavitharana 54

Map Translation Using Geo-tagged Social Media
Sunyou Lee, Taesung Lee and Seung-won Hwang 59

Predicting Romanian Stress Assignment
Alina Maria Ciobanu, Anca Dinu and Liviu Dinu 64

Passive-Aggressive Sequence Labeling with Discriminative Post-Editing for Recognising Person Entities in Tweets
Leon Derczynski and Kalina Bontcheva 69

Accelerated Estimation of Conditional Random Fields using a Pseudo-Likelihood-inspired Perceptron Variant
Teemu Ruokolainen, Miikka Silfverberg, mikko kurimo and Krister Linden 74

Deterministic Word Segmentation Using Maximum Matching with Fully Lexicalized Rules
Manabu Sassano 79

Painless Semi-Supervised Morphological Segmentation using Conditional Random Fields
Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja and mikko kurimo 84

Inference of Phrase-Based Translation Models via Minimum Description Length
Jesús González-Rubio and Francisco Casacuberta 90

Chinese Native Language Identification
Shervin Malmasi and Mark Dras 95

Unsupervised Parsing for Generating Surface-Based Relation Extraction Patterns
Jens Illig, Benjamin Roth and Dietrich Klakow 100

Automatic Selection of Reference Pages in Wikipedia for Improving Targeted Entities Disambiguation
Takuya Makino 106

Using a Random Forest Classifier to Compile Bilingual Dictionaries of Technical Terms from Comparable Corpora
Georgios Kontonatsios, Ioannis Korkontzelos, Jun'ichi Tsujii and Sophia Ananiadou 111

Comparing methods for deriving intensity scores for adjectives
Josef Ruppenhofer, Michael Wiegand and Jasper Brandes 117

Bayesian Word Alignment for Massively Parallel Texts
Robert Östling 123

Acquiring a Dictionary of Emotion-Provoking Events
Hoa Trong Vu, Graham Neubig, Sakriani Sakti, Tomoki Toda and Satoshi Nakamura 128

Chinese Temporal Tagging with HeidelTime
Hui Li, Jannik Strötgen, Julian Zell and Michael Gertz 133

A Probabilistic Approach to Persian Ezafe Recognition
Habibollah Asghari, Jalal Maleki and Heshaam Faili 138

Converting Russian Dependency Treebank to Stanford Typed Dependencies Representation
Janna Lipenkova and Milan Souček 143

Integrating an Unsupervised Transliteration Model into Statistical Machine Translation
Nadir Durrani, Hassan Sajjad, Hieu Hoang and Philipp Koehn 148

Improving Dependency Parsers with Supertags
Hiroki Ouchi, Kevin Duh and Yuji Matsumoto 154

Improving Dependency Parsers using Combinatory Categorial Grammar
Bharat Ram Ambati, Tejaswini Deoskar and Mark Steedman 159

Fast and Accurate Unlexicalized Parsing via Structural Annotations
Maximilian Schlund, Michael Luttenberger and Javier Esparza 164

Data Driven Language Transfer Hypotheses
Ben Swanson and Eugene Charniak 169

Simple and Effective Approach for Consistent Training of Hierarchical Phrase-based Translation Models
Stephan Peitz, David Vilar and Hermann Ney 174

Some Experiments with a Convex IBM Model 2
Andrei Simion, Michael Collins and Cliff Stein 180

Active Learning for Post-Editing Based Incrementally Retrained MT
Aswarth Abhilash Dara, Josef van Genabith, Qun Liu, John Judge and Antonio Toral 185

Analysis and Prediction of Unalignable Words in Parallel Text
Frances Yung, Kevin Duh and Yuji Matsumoto 190

Enhancing Authorship Attribution By Utilizing Syntax Tree Profiles
Michael Tschuggnall and Günther Specht 195

Multi-Domain Sentiment Relevance Classification with Automatic Representation Learning
Christian Scheible and Hinrich Schütze 200

A New Entity Salience Task with Millions of Training Examples
Jesse Dunietz and Daniel Gillick 205

Finding middle ground? Multi-objective Natural Language Generation from time-series data
Dimitra Gkatzia, Helen Hastie and Oliver Lemon 210

One Sense per Tweeter ... and Other Lexical Semantic Tales of Twitter
Spandana Gella, Paul Cook and Timothy Baldwin 215

Zero subject detection for Polish
Mateusz Kopeć 221

Crowdsourcing Annotation of Non-Local Semantic Roles
Parvin Sadat Feizabadi and Sebastian Padó 226

Coreference Resolution Evaluation for Higher Level Applications
Don Tuggener 231

Efficient Online Summarization of Microblogging Streams
Andrei Olariu 236

Conference Program

Monday, April 28

Session S1A: (14:00-15:00) Information Retrieval and Text Mining

Easy Web Search Results Clustering: When Baselines Can Reach State-of-the-Art Algorithms
Jose G. Moreno and Gaël Dias

Propagation Strategies for Building Temporal Ontologies
Mohammed Hasanuzzaman, Gaël Dias, Stéphane Ferrari and Yann Mathet

Chinese Open Relation Extraction for Knowledge Acquisition
Yuen-Hsien Tseng, Lung-Hao Lee, Shu-Yen Lin, Bo-Shun Liao, Mei-Jun Liu, Hsin-Hsi Chen, Oren Etzioni and Anthony Fader

Temporal Text Ranking and Automatic Dating of Texts
Vlad Niculae, Marcos Zampieri, Liviu Dinu and Alina Maria Ciobanu

Session S1B: (14:00-15:00) Semantics

Measuring the Similarity between Automatically Generated Topics
Nikolaos Aletras and Mark Stevenson

Projecting the Knowledge Graph to Syntactic Parsing
Andrea Gesmundo and Keith Hall

A Vague Sense Classifier for Detecting Vague Definitions in Ontologies
Panos Alexopoulos and John Pavlopoulos

Chasing Hypernyms in Vector Spaces with Entropy
Enrico Santus, Alessandro Lenci, Qin Lu and Sabine Schulte im Walde

Monday, April 28 (continued)

Session S1C: (14:00-15:00) Spoken Language Processing and Machine Translation

Tight Integration of Speech Disfluency Removal into SMT
Eunah Cho, Jan Niehues and Alex Waibel

Non-Monotonic Parsing of Fluent Umm I mean Disfluent Sentences
Mohammad Sadegh Rasooli and Joel Tetreault

Lightly-Supervised Word Sense Translation Error Detection for an Interactive Conversational Spoken Language Translation System
Dennis Mehay, Sankaranarayanan Ananthakrishnan and Sanjika Hewavitharana

Map Translation Using Geo-tagged Social Media
Sunyou Lee, Taesung Lee and Seung-won Hwang

Session S1D: (14:00-15:00) Machine Learning and Sequence Labeling

Predicting Romanian Stress Assignment
Alina Maria Ciobanu, Anca Dinu and Liviu Dinu

Passive-Aggressive Sequence Labeling with Discriminative Post-Editing for Recognising Person Entities in Tweets
Leon Derczynski and Kalina Bontcheva

Accelerated Estimation of Conditional Random Fields using a Pseudo-Likelihood-inspired Perceptron Variant
Teemu Ruokolainen, Miikka Silfverberg, mikko kurimo and Krister Linden

Deterministic Word Segmentation Using Maximum Matching with Fully Lexicalized Rules
Manabu Sassano

Monday, April 28 (continued)

SP Posters

Painless Semi-Supervised Morphological Segmentation using Conditional Random Fields
Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja and mikko kurimo

Inference of Phrase-Based Translation Models via Minimum Description Length
Jesús González-Rubio and Francisco Casacuberta

Chinese Native Language Identification
Shervin Malmasi and Mark Dras

Unsupervised Parsing for Generating Surface-Based Relation Extraction Patterns
Jens Illig, Benjamin Roth and Dietrich Klakow

Automatic Selection of Reference Pages in Wikipedia for Improving Targeted Entities Disambiguation
Takuya Makino

Using a Random Forest Classifier to Compile Bilingual Dictionaries of Technical Terms from Comparable Corpora
Georgios Kontonatsios, Ioannis Korkontzelos, Jun'ichi Tsujii and Sophia Ananiadou

Comparing methods for deriving intensity scores for adjectives
Josef Ruppenhofer, Michael Wiegand and Jasper Brandes

Bayesian Word Alignment for Massively Parallel Texts
Robert Östling

Acquiring a Dictionary of Emotion-Provoking Events
Hoa Trong Vu, Graham Neubig, Sakriani Sakti, Tomoki Toda and Satoshi Nakamura

Chinese Temporal Tagging with HeidelTime
Hui Li, Jannik Strötgen, Julian Zell and Michael Gertz

A Probabilistic Approach to Persian Ezafe Recognition
Habibollah Asghari, Jalal Maleki and Heshaam Faili

Converting Russian Dependency Treebank to Stanford Typed Dependencies Representation
Janna Lipenkova and Milan Souček

Monday, April 28 (continued)

Integrating an Unsupervised Transliteration Model into Statistical Machine Translation
Nadir Durrani, Hassan Sajjad, Hieu Hoang and Philipp Koehn

Tuesday, April 29

Session S2A: (15:00-16:00) Parsing

Improving Dependency Parsers with Supertags
Hiroki Ouchi, Kevin Duh and Yuji Matsumoto

Improving Dependency Parsers using Combinatory Categorial Grammar
Bharat Ram Ambati, Tejaswini Deoskar and Mark Steedman

Fast and Accurate Unlexicalized Parsing via Structural Annotations
Maximilian Schlund, Michael Luttenberger and Javier Esparza

Data Driven Language Transfer Hypotheses
Ben Swanson and Eugene Charniak

Session S2B: (15:00-16:00) Machine Translation

Simple and Effective Approach for Consistent Training of Hierarchical Phrase-based Translation Models
Stephan Peitz, David Vilar and Hermann Ney

Some Experiments with a Convex IBM Model 2
Andrei Simion, Michael Collins and Cliff Stein

Active Learning for Post-Editing Based Incrementally Retrained MT
Aswarth Abhilash Dara, Josef van Genabith, Qun Liu, John Judge and Antonio Toral

Analysis and Prediction of Unalignable Words in Parallel Text
Frances Yung, Kevin Duh and Yuji Matsumoto

Tuesday, April 29 (continued)

Session S2C: (15:00-16:00) Sentiment Analysis and Generation

Enhancing Authorship Attribution By Utilizing Syntax Tree Profiles
Michael Tschuggnall and Günther Specht

Multi-Domain Sentiment Relevance Classification with Automatic Representation Learning
Christian Scheible and Hinrich Schütze

A New Entity Salience Task with Millions of Training Examples
Jesse Dunietz and Daniel Gillick

Finding middle ground? Multi-objective Natural Language Generation from time-series data
Dimitra Gkatzia, Helen Hastie and Oliver Lemon

Session S2D: (15:00-16:00) Discourse and Semantics

One Sense per Tweeter ... and Other Lexical Semantic Tales of Twitter
Spandana Gella, Paul Cook and Timothy Baldwin

Zero subject detection for Polish
Mateusz Kopeć

Crowdsourcing Annotation of Non-Local Semantic Roles
Parvin Sadat Feizabadi and Sebastian Padó

Coreference Resolution Evaluation for Higher Level Applications
Don Tuggener

Wednesday, April 30

Best short paper

Efficient Online Summarization of Microblogging Streams
Andrei Olariu

Easy Web Search Results Clustering: When Baselines Can Reach State-of-the-Art Algorithms

Jose G. Moreno
Normandie University
UNICAEN, GREYC CNRS
F-14032 Caen, France
jose.moreno@unicaen.fr

Gaël Dias
Normandie University
UNICAEN, GREYC CNRS
F-14032 Caen, France
gael.dias@unicaen.fr

Abstract

This work discusses the evaluation of baseline algorithms for Web search results clustering. An analysis is performed over frequently used baseline algorithms and standard datasets. Our work shows that competitive results can be obtained by either fine tuning or performing cascade clustering over well-known algorithms. In particular, the latter strategy can lead to a scalable and real-world solution, which evidences comparative results to recent text-based state-of-the-art algorithms.

1 Introduction

Visualizing Web search results remains an open problem in Information Retrieval (IR). For example, in order to deal with ambiguous or multi-faceted queries, many works present Web page results using groups of correlated contents instead of long flat lists of relevant documents. Among existing techniques, Web Search Results Clustering (SRC) is a commonly studied area, which consists in clustering "on-the-fly" Web page results based on their Web snippets. Therefore, many works have been recently presented including task adapted clustering (Moreno et al., 2013), meta clustering (Carpineto and Romano, 2010) and knowledge-based clustering (Scaiella et al., 2012).

Evaluation is also a hot topic both in Natural Language Processing (NLP) and IR. Within the specific case of SRC, different metrics have been used such as F_1-measure (F_1), $kSSL$[1] and F_{b^3}-measure (F_{b^3}) over different standard datasets: ODP-239 (Carpineto and Romano, 2010) and Moresque (Navigli and Crisafulli, 2010). Unfortunately, comparative results are usually biased as baseline algorithms are run with default parameters whereas proposed methodologies are usually tuned to increase performance over the studied datasets. Moreover, evaluation metrics tend to correlate with the number of produced clusters.

In this paper, we focus on deep understanding of the evaluation task within the context of SRC. First, we provide the results of baseline algorithms with their best parameter settings. Second, we show that a simple cascade strategy of baseline algorithms can lead to a scalable and real-world solution, which evidences comparative results to recent text-based algorithms. Finally, we draw some conclusions about evaluation metrics and their bias to the number of output clusters.

2 Related Work

Search results clustering is an active research area. Two main streams have been proposed so far: text-based strategies such as (Hearst and Pedersen, 1996; Zamir and Etzioni, 1998; Zeng et al., 2004; Osinski et al., 2004; Carpineto and Romano, 2010; Carpineto et al., 2011; Moreno et al., 2013) and knowledge-based ones (Ferragina and Gulli, 2008; Scaiella et al., 2012; Di Marco and Navigli, 2013). Successful results have been obtained by recent works compared to STC (Zamir and Etzioni, 1998) and LINGO (Osinski et al., 2004) which provide publicly available implementations, and as a consequence, are often used as state-of-the-art baselines. On the one hand, STC proposes a monothetic methodology which merges base clusters with high string overlap relying on suffix trees. On the other hand, LINGO is a polythetic solution which reduces a term-document matrix using single value decomposition and assigns documents to each discovered latent topic.

All solutions have been evaluated on different datasets and evaluation measures. The well-known F_1 has been used as the standard evaluation metric. More recently, (Carpineto and Romano,

[1] This metric is based on subjective label evaluation and as such is out of the scope of this paper.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 1–5,
Gothenburg, Sweden, April 26-30 2014.

	Moresque								ODP-239							
	F_1				F_{b^3}				F_1				F_{b^3}			
Algo.	Stand.	k	Tuned	k	Stand.	k	Tuned	k	Stand.	k	Tuned	k	Stand.	k	Tuned	k
STC	0.4550	12.7	0.6000	2.9	0.4602	12.7	0.4987	2.9	0.3238	12.4	0.3350	3.0	0.4027	12.4	0.4046	14.5
LINGO	0.3258	26.7	**0.6034**	3.0	0.3989	26.7	**0.5004**	5.8	0.2029	27.7	0.3320	3.0	0.3461	27.7	**0.4459**	8.7
BiKm	0.3165	9.7	0.5891	2.1	0.3145	9.7	0.4240	2.1	0.1995	12.1	**0.3381**	2.2	0.3074	12.1	0.3751	2.2
Random	-	-	0.5043	2	-	-	0.3548	2	-	-	0.2980	2	-	-	0.3212	2

Table 1: Standard, Tuned and Random Results for Moresque and ODP-239 datasets.

2010) evidenced more complete results with the general definition of the F_β-measure for $\beta = \{1, 2, 5\}$, (Navigli and Crisafulli, 2010) introduced the Rand Index metric and (Moreno et al., 2013) used F_{b^3} introduced by (Amigó et al., 2009) as a more adequate metric for clustering.

Different standard datasets have been built such as AMBIENT[2] (Carpineto and Romano, 2009), ODP-239[3] (Carpineto and Romano, 2010) and Moresque[4] (Navigli and Crisafulli, 2010). ODP-239, an improved version of AMBIENT, is based on DMOZ[5] where each query, over 239 ones, is a selected category in DMOZ and its associated subcategories are considered as the respective cluster results. The small text description included in DMOZ is considered as a Web snippet. Moresque is composed by 114 queries selected from a list of ambiguous Wikipedia entries. For each query, a set of Web results have been collected from a commercial search engine and manually classified into the disambiguation Wikipedia pages which form the reference clusters.

In Table 2, we report the results obtained so far in the literature by text-based and knowledge-based strategies for the standard F_1 over ODP-239 and Moresque datasets.

		F_1	
		ODP239	Moresque
Text	STC	0.324	0.455
	LINGO	0.273	0.326
	(Carpineto and Romano, 2010)	0.313	-
	(Moreno et al., 2013)	0.390	0.665
Know.	(Scaiella et al., 2012)	0.413	-
	(Di Marco and Navigli, 2013)	-	0.7204*

Table 2: State-of-the-art Results for SRC. (*) The result of (Di Marco and Navigli, 2013) is based on a reduced version of AMBIENT + Moresque.

3 Baseline SRC Algorithms

Newly proposed algorithms are usually tuned towards their maximal performance. However, the results of baseline algorithms are usually run with default parameters based on available implementations. As such, no conclusive remarks can be drawn knowing that tuned versions might provide improved results.

In particular, available implementations[6] of STC, LINGO and the Bisection K-means (BiKm) include a fixed stopping criterion. However, it is well-known that tuning the number of output clusters may greatly impact the clustering performance. In order to provide fair results for baseline algorithms, we evaluated a k-dependent[7] version for all baselines. We ran all algorithms for $k = 2..20$ and chose the best result as the "optimal" performance. Table 1 sums up results for all the baselines in their different configurations and shows that tuned versions outperform standard (available) ones both for F_1 and F_{b^3} over ODP-239 and Moresque.

4 Cascade SRC Algorithms

In the previous section, our aim was to claim that tunable versions of existing baseline algorithms might evidence improved results when faced to the ones reported in the literature. And these values should be taken as the "real" baseline results within the context of controllable environments. However, exploring all the parameter space is not an applicable solution in a real-world situation where the reference is unknown. As such, a stopping criterion must be defined to adapt to any dataset distribution. This is the particular case for the standard implementations of STC and LINGO.

Previous results (Carpineto and Romano, 2010) showed that different SRC algorithms provide different results and hopefully complementary ones. For instance, STC demonstrates high recall and low precision, while LINGO inversely evidences high precision for low recall. Iteratively applying baseline SRC algorithms may thus lead to improved results by exploiting each algorithm's strengths.

[2] http://credo.fub.it/ambient/ [Last acc.: Jan., 2014]

[3] http://credo.fub.it/odp239/ [Last acc.: Jan., 2014]

[4] http://lcl.uniroma1.it/moresque/ [Last acc.: Jan., 2014]

[5] http://www.dmoz.org [Last acc.: Jan., 2014]

[6] http://carrot2.org [Last acc.: Jan., 2014]

[7] Carrot2 parameters *maxClusters*, *desiredClusterCountBase* and *clusterCount* were used to set k value.

In a cascade strategy, we first cluster the initial set of Web page snippets with any SRC algorithm. Then, the input of the second SRC algorithm is the set of meta-documents built from the documents belonging to the same cluster[8]. Finally, each clustered meta-document is mapped to the original documents generating the final clusters. This process can iteratively be applied, although we only consider two-level cascade strategies in this paper.

This strategy can be viewed as an easy, reproducible and parameter free baseline SRC implementation that should be compared to existing state-of-the-art algorithms. Table 3 shows the results obtained with different combinations of SRC baseline algorithms for the cascade strategy both for F_1 and F_{b^3} over ODP-239 and Moresque. The "Stand." column corresponds to the performance of the cascade strategy and k to the automatically obtained number of clusters. Results show that the combination STC-STC achieves the best performance overall for the F_1 and STC-LINGO is the best combination for the F_{b^3} in both datasets.

In order to provide a more complete evaluation, we included in column "Equiv." the performance that could be obtained by the tunable version of each single baseline algorithm based on the same k. Interestingly, the cascade strategy outperforms the tunable version for any k for F_1 but fails to compete (not by far) with F_{b^3}. This issue will be discussed in the next section.

5 Discussion

In Table 1, one can see that when using the tuned version and evaluating with F_1, the best performance for each baseline algorithm is obtained for the same number of output clusters independently of the dataset (i.e. around 3 for STC and LINGO and 2 for BiKm). As such, a fast conclusion would be that the tuned versions of STC, LINGO and BiKm are strong baselines as they show similar behaviour over datasets. Then, in a realistic situation, k might be directly tuned to these values.

However, when comparing the output number of clusters based on the best F_1 value to the reference number of clusters, a huge difference is evidenced. Indeed, in Moresque, the ground-truth average number of clusters is 6.6 and exactly 10 in ODP-239. Interestingly, F_{b^3} shows more accurate values for the number of output clusters for the best tuned baseline performances. In particular, the best F_{b^3} results are obtained for LINGO with 5.8 clusters for Moresque and 8.7 clusters for ODP-239 which most approximate the ground-truths.

In order to better understand the behaviour of each evaluation metric (i.e. F_β and F_{b^3}) over different k values, we experienced a uniform random clustering over Moresque and ODP-239. In Figure 1(c), we illustrate these results. The important issue is that F_β is more sensitive to the number of output clusters than F_{b^3}. On the one hand, all F_β measures provide best results for $k = 2$ and a random algorithm could reach F_1=0.5043 for Moresque and F_1=0.2980 for ODP-239 (see Table 1), thus outperforming almost all standard implementations of STC, LINGO and BiKm for both datasets. On the other hand, F_{b^3} shows that most standard baseline implementations outperform the random algorithm.

Moreover, in Figures 1(a) and 1(b), we illustrate the different behaviours between F_1 and F_{b^3} for $k = 2..20$ for both standard and tuned versions of STC, LINGO and BiKm. One may clearly see that F_{b^3} is capable to discard the algorithm (BiKm) which performs worst in the standard version while this is not the case for F_1. And, for LINGO, the optimal performances over Moresque and ODP-239 are near the ground-truth number of clusters while this is not the case for F_1 which evidences a decreasing tendency when k increases.

In section 4, we showed that competitive results could be achieved with a cascade strategy based on baseline algorithms. Although results outperform standard and tunable baseline implementations for F_1, it is wise to use F_{b^3} to better evaluate the SRC task, based on our previous discussion. In this case, the best values are obtained by STC-LINGO with F_{b^3}=0.4980 for Moresque and F_{b^3}=0.4249 for ODP-239, which highly approximate the values reported in (Moreno et al., 2013): F_{b^3}=0.490 (Moresque) and F_{b^3}=0.452 (ODP-239). Additionally, when STC is performed first and LINGO later the cascade algorithm scale better due to LINGO and STC scaling properties[9].

6 Conclusion

This work presents a discussion about the use of baseline algorithms in SRC and evaluation met-

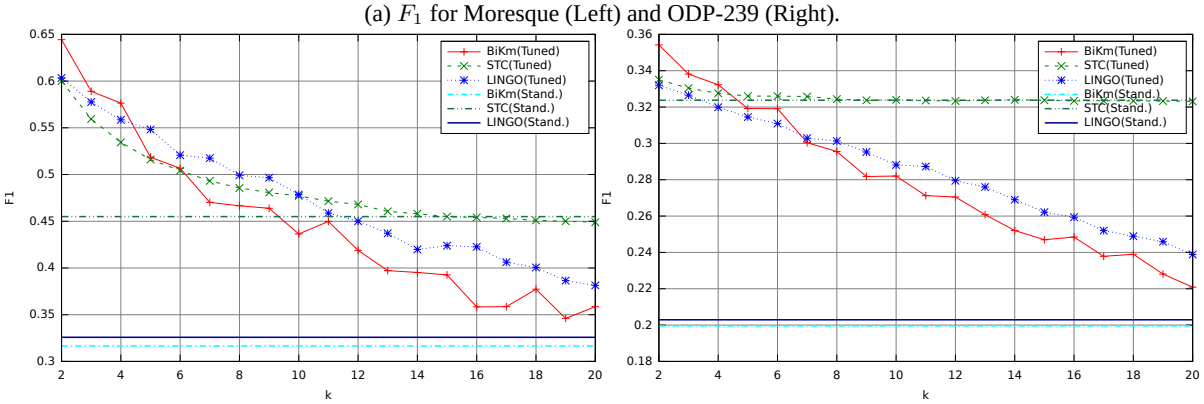
[8] Fused using concatenation of strings.

[9] http://carrotsearch.com/lingo3g-comparison [Last acc.: Jan., 2014]

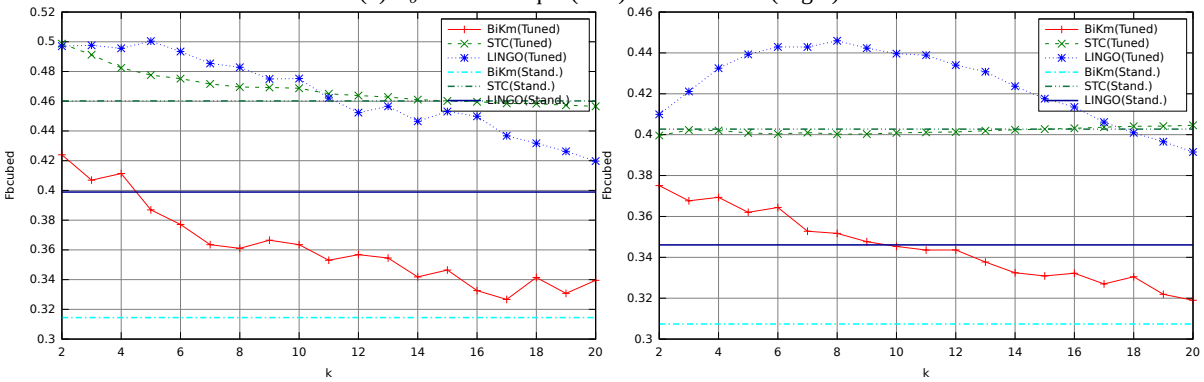
		Moresque						ODP-239					
		F_1			F_{b^3}			F_1			F_{b^3}		
Level 1	Level 2	Stand.	Equiv.	k	Stand.	Equiv.	k	Stand.	Equiv.	k	Stand.	Equiv.	k
STC	STC	**0.6145**	0.5594	3.1	0.4550	**0.4913**	3.1	**0.3629**	0.3304	3.2	0.3982	0.4023	3.2
	LINGO	0.5611	0.4932	7.3	**0.4980**	0.4716	7.3	0.3624	0.3258	6.9	**0.4249**	0.4010	6.9
	BiKm	0.5413	0.5160	4.5	0.4395	0.4776	4.5	0.3319	0.3276	4.3	0.3845	0.4020	4.3
LINGO	STC	0.5696	0.5176	6.7	0.4602	0.4854	6.7	0.3457	0.3029	7.2	0.4229	**0.4429**	7.2
	LINGO	0.4629	0.4371	13.7	0.4447	0.4566	13.7	0.2789	0.2690	13.6	0.3931	0.4237	13.6
	BiKm	0.4038	0.4966	8.6	0.3801	0.4750	8.6	0.2608	0.2953	8.5	0.3510	0.4423	8.5
BiKm	STC	0.5873	**0.5891**	2.7	0.4144	0.4069	2.7	0.3425	0.3381	2.7	0.3787	0.3677	2.7
	LINGO	0.4773	0.5186	5.4	0.3832	0.3869	5.4	0.2819	0.3191	6.3	0.3546	0.3644	6.3
	BiKm	0.4684	0.5764	3.5	0.3615	0.4114	3.5	0.2767	**0.3322**	4.3	0.3328	0.3693	4.3

Table 3: Cascade Results for Moresque and ODP-239 datasets.

(a) F_1 for Moresque (Left) and ODP-239 (Right).



(b) F_{b^3} for Moresque (Left) and ODP-239 (Right).



(c) Evaluation Metrics for Random Clustering for Moresque (Left) and ODP-239 (Right).

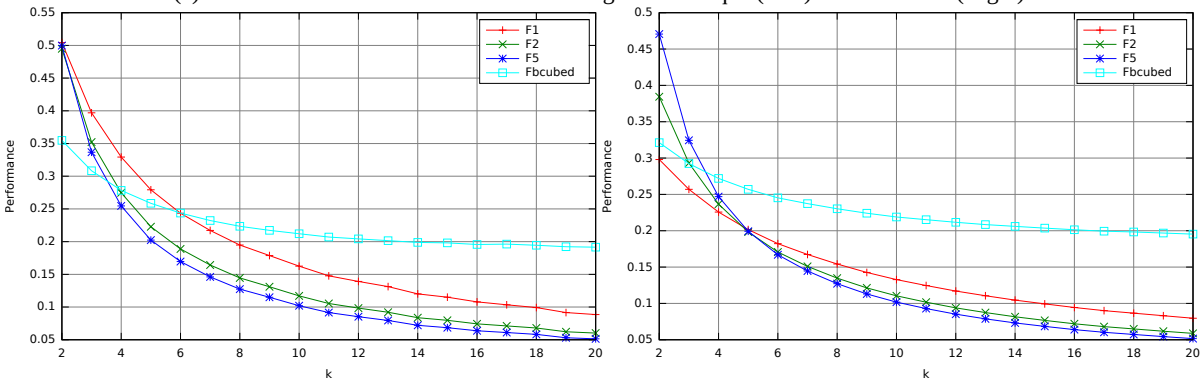


Figure 1: F_1 and F_{b^3} for Moresque and ODP-239 for Standard, Tuned and Random Clustering.

rics. Our experiments show that F_{b^3} seems more adapted to evaluate SRC systems than the commonly used F_1 over the standard datasets available so far. New baseline values which approximate state-of-the-art algorithms in terms of clustering performance can also be obtained by an easy, reproducible and parameter free implementation (the cascade strategy) and could be considered as the "new" baseline results for future works.

References

E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. 2009. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12(4):461–486.

C. Carpineto and G. Romano. 2009. Mobile information retrieval with search results clustering : Prototypes and evaluations. *Journal of the American Society for Information Science*, 60:877–895.

C. Carpineto and G. Romano. 2010. Optimal meta search results clustering. In *33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 170–177.

C. Carpineto, M. D'Amico, and A. Bernardini. 2011. Full discrimination of subtopics in search results with keyphrase-based clustering. *Web Intelligence and Agent Systems*, 9(4):337–349.

A. Di Marco and R. Navigli. 2013. Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, 39(3):709–754.

P. Ferragina and A. Gulli. 2008. A personalized search engine based on web-snippet hierarchical clustering. *Software: Practice and Experience*, 38(2):189–225.

M.A. Hearst and J.O. Pedersen. 1996. Re-examining the cluster hypothesis: Scatter/gather on retrieval results. In *19th Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 76–84.

J.G. Moreno, G. Dias, and G. Cleuziou. 2013. Post-retrieval clustering using third-order similarity measures. In *51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 153–158.

R. Navigli and G. Crisafulli. 2010. Inducing word senses to improve web search result clustering. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 116–126.

S. Osinski, J. Stefanowski, and D. Weiss. 2004. Lingo: Search results clustering algorithm based on singular value decomposition. In *Intelligent Information Systems Conference (IIPWM)*, pages 369–378.

U. Scaiella, P. Ferragina, A. Marino, and M. Ciaramita. 2012. Topical clustering of search results. In *5th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 223–232.

O. Zamir and O. Etzioni. 1998. Web document clustering: A feasibility demonstration. In *21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 46–54.

H.J. Zeng, Q.C. He, Z. Chen, W.Y. Ma, and J. Ma. 2004. Learning to cluster web search results. In *27th Annual International Conference on Research and Development in Information Retrieval (SIGIR)*, pages 210–217.

Propagation Strategies for Building Temporal Ontologies

Md. Hasanuzzaman
Normandie University
GREYC UMR 6072
Caen, France

Gaël Dias
Normandie University
GREYC UMR 6072
Caen, France

Stéphane Ferrari
Normandie University
GREYC UMR 6072
Caen, France

Yann Mathet
Normandie University
GREYC UMR 6072
Caen, France

Abstract

In this paper, we propose to build temporal ontologies from WordNet. The underlying idea is that each synset is augmented with its temporal connotation. For that purpose, temporal classifiers are iteratively learned from an initial set of time-sensitive synsets and different propagation strategies to give rise to different TempoWordNets.

1 Introduction

Temporality has recently received increased attention in Natural Language Processing (NLP) and Information Retrieval (IR). Initial works have been proposed in NLP and are exhaustively summarized in (Mani et al., 2005). More recently, the introduction of the TempEval task (Verhagen et al., 2009) in the Semantic Evaluation workshop series has clearly established the importance of time to deal with different NLP tasks. The ultimate aim of research in this area is the automatic identification of temporal expressions (timexes), events and temporal relations within a text in the TimeML format (Pustejovsky et al., 2005).

In IR, the time dimension has also received particular attention for the past few years. According to (Metzger, 2007), time is one of the key five aspects that determine a document credibility besides relevance, accuracy, objectivity and coverage. So, the value of information or its quality is intrinsically time-dependent. As a consequence, a new reasearch field called Temporal Information Retrieval (T-IR) has emerged (Alonso et al., 2011) and deals with all classical IR tasks such as crawling (Kulkarni et al., 2011), indexing (Anand et al., 2012) or ranking (Kanhabua et al., 2011) from the time viewpoint.

However, both NLP and IR evidence the lack of temporal lexical resources. For example, automatic temporal ordering of events in text is usually performed via various linguistic mechanisms including the use of time expressions such as "before","after" or "during" that explicitly assert a temporal relation. In particular, (Derczynski and Gaizauskas, 2012) investigate the role of temporal signals in temporal relation extraction over the TimeBank annotated corpus. However, the list of such expressions is limited. From the IR viewpoint, most methodologies rely on the presence of explicit timexes and hardly bridge the gap when no explicit mention of time is available. One recent exception is proposed in (Jatowt et al., 2013) where text time-tagging is seen as a classification task, but no use of specific temporal clues is introduced or proposed.

Inspired by SentiWordNet (Esuli and Sebastiani, 2006), we propose to introduce the temporal connotation of each synset in WordNet (Miller, 1995) by iteratively learning temporal classifiers from an initial set of time-sensitive synsets and a given propagation strategy. As such, each synset is automatically time-tagged with four dimensions i.e. *atemporal*, *past*, *present* and *future*, thus giving rise to different TempoWordNets depending on the propagation strategy.

TempoWordNets are evaluated both manually and automatically. First, results show that manual annotation of time-tagged synsets is a hard task for humans. Second, automatic evaluation based on sentence temporal classification shows that the introduction of time-augmented lexical knowledge bases (TempoWordNets) allows 3.9% improvements of F_1-measure against the vector space model representation and 4.2% against the semantic vector space model obtained with the existing WordNet time subtree.

2 Related Work

A great deal of works have been proposed in temporal NLP. Most recent studies have been developed in the context of the TempEval evaluation contests, which were initiated by (Verhagen et al.,

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 6–11,
Gothenburg, Sweden, April 26-30 2014.

2007). TempEval was initially divided into three challenges: (A) identifying temporal relations between events and time expressions, (B) identifying temporal relations between events and the document creation time and (C) identifying the temporal relations between contiguous pairs of matrix verbs. In TempEval-2 (Pustejovsky and Verhagen, 2009), the best performing systems were based on conditional random fields mixed with parsing methodologies (UzZaman and Allen, 2010). More recently, in TempEval-3 (UzZaman et al., 2013), new systems have been performing at high level of performance for all three tasks such as the rule-based multilingual temporal tagger Heideltime (Strötgen and Gertz, 2013). In IR, the work of (Baeza-Yates, 2005) defines the foundations of T-IR. Since, research have been tackling several topics such as query understanding (Metzler et al., 2009), temporal snippets generation (Alonso et al., 2007), temporal ranking (Kanhabua et al., 2011), temporal clustering (Alonso et al., 2009), future retrieval (Radinsky and Horvitz, 2013) or temporal image retrieval (Dias et al., 2012).

As expressed in (Strötgen and Gertz, 2013), time taggers usually contain pattern files with words and phrases, which are typically used to express temporal expressions in a given language (e.g. names of months). In fact, most temporal NLP tasks rely on a time-sensitive vocabulary. On the contrary, T-IR systems usually do not use information about time in language although they could benefit from it when facing the recurrent problem of missing explicit timexes.

WordNet is a good place to start to find time-sensitive concepts. Indeed, one can list a set of 21 temporal synsets by iteratively following the hyponym relation from the concept of time (synset # 00028270) represented by the following gloss: *the continuum of experience in which events pass from the future through the present to the past*. However, likewise the tennis problem evidenced in (Fellbaum, 1998), most temporal words are not under the concept of time. For example, concepts such as "prediction", "remember", "ancient" or "fresh" clearly have a time dimension although they are not listed under the time subtree of WordNet. Based on the intial ideas of (Moens and Steedman, 1987) on temporal ontologies and inspired by SentiWordNet (Esuli and Sebastiani, 2006), we propose to enrich all WordNet synsets with their temporal connotation.

3 TempoWordNet as SentiWordNet

In (Dias et al., 2014), we first proposed to build TempoWordNet based on the idea of (Esuli and Sebastiani, 2006) for SentiWordNet. Each synset is automatically time-tagged with four dimensions i.e. *atemporal*, *past*, *present* and *future* by performing a two-step process.

A first temporal classifier is built based on a set of manually selected seed synsets and their corresponding glosses tagged as *past*, *present* and *future*. This process is then iterated based on the repetitive lexico-semantic expansion[1] of the initial seeds lists until cross-validation accuracy drops. This first step results in a three-class temporal classifier and an expanded list of temporal synset candidates.

A second temporal classifier is then learned to time-tag synsets as *atemporal* or *temporal*. This process is obtained by taking the final list of expanded seed synsets from the previous learning problem and randomly choosing a balanced number atemporal synsets. A 10-fold cross-validation is then used to learn the model.

TempoWordNet is finally obtained by (1) classifying all WordNet synsets as *atemporal* or *temporal* with the second classifier and (2) the resulting temporal synsets are tagged as *past*, *present* and *future* by the first classifier. This step is detailed in (Dias et al., 2014) and all materials can be found at http://tempowordnet.greyc.fr.

4 Diversified Expansion Strategies

The initial strategy proposed in the previous section evidences a clear lack. As the expansion process is semantically driven, the temporal connotation is highly depend on the initial seeds lists and as a consequence may not spread over a wide range of concepts in WordNet. As such, we propose two different strategies of expansion: (1) the probabilistic expansion and (2) the hybrid (probabilistic combined with semantic) expansion.

Probabilistic Expansion: We first learn a *temporal* vs. *atemporal* classifier based on the initial hand-crafted set of seeds proposed in (Dias et al., 2014). In particular, the seeds defined as *past*, *present* and *future* are markers of temporality, while the list of *atemporal* synsets is the obvious counterpart. Based on this list of *tempo-*

[1] Only exisiting lexico-semantic links in WordNet are used to propagate the temporal connotation.

ral and *atemporal* synsets, a 10-fold cross validation process is performed to learn the *temporal* vs. *atemporal* model, which is used to time-tag the whole WordNet. The synsets (or glosses) with highest *temporal* and *atemporal* values in WordNet are then used for the expansion process of the seeds lists. The process is iteratively performed and stops when accuracy drops.

After building the *temporal* vs. *atemporal* classifier, WordNet is divided into two subsets: *temporal* synsets and *atemporal* ones. In order to fine tune the *temporal* part of WordNet, we learn a three-class classifier (i.e. *past*, *present* and *future*) based on the initial *past*, *present* and *future* seeds lists and the probabilistic expansion exclusively[2] within the temporal part of WordNet. So, a 10-fold cross validation process is iteratively performed until accuracy drops.

The results of the probabilistic expansion are presented in Table 1 and Table 2, when the expansion is based on the maximum probability value[3].

Steps	1	2	3
Precision	87.3	**100**	100
Recall	86.7	**100**	100
F_1-measure	86.9	**100**	100

Table 1: Cross validation for *temporal* vs. *atemporal* at each iteration. Probabilistic Expansion.

Steps	1	2	3
Precision	80.0	**99.7**	99.6
Recall	80.1	**99.7**	99.6
F_1-measure	80.0	**99.7**	99.6

Table 2: Cross validation for *past*, *present* and *future* at each iteration. Probabilistic Expansion.

Note that in our experiment, Support Vector Machines (SVM) with a linear kernel[4] over the vector space model representation of the synsets (i.e. each synset is represented by its gloss encoded as a vector of unigrams weighted by their frequency) have been used to classify all the synsets of WordNet. The results show that in both cases the expansion process stops at iteration 2.

[2]Only temporal synsets are classified as *past*, *present* or *future* and used for the expansion process. Note that unbalanced sets can be formed.

[3]That means that all the synsets getting the highest value produced by the classifier are used to expand the initial seeds lists.

[4]We used the Weka implementation SMO with default parameters.

Hybrid Expansion: Choosing synsets from WordNet with highest probability assigned by a classifier learned on the glosses of initial seeds lists can lead to the well-known semantic shift problem. So, the idea of the hybrid expansion is to control the expansion process so that the most probable time-sensitive synsets are also chosen based on their semantic distance with the expanded seed synsets at the previous iteration. The process is straightforward when compared to the probabilistic expansion.

First, a two-class (*temporal* vs. *atemporal*) text classifier is trained based on the glosses of each synsets contained in the initial seed lists to classify all the synsets of WordNet. Thereafter, WordNet synsets with highest probability are selected as candidates for expansion. From these candidates, only the ones that present the maximum semantic similarity to the previous seeds lists are chosen for expansion. Note that the semantic similarity is calculated between the candidate synset and all synsets in the previous expanded seeds lists. Once candidates for expansion have been chosen, a 10-fold cross validation process is iteratively performed until accuracy becomes steady.

Second, a three-class (*past*, *present* and *future*) classifier is learned over the *temporal* part of WordNet with the hybrid expansion process in the same exact manner as explained for the previous probabilistic expansion. Results for the expansion process are presented in the Table 3 and Table 4 for the same experimental setups as for the probabilistic expansion and using the (Leacock et al., 1998) semantic similarity measure[5].

Steps	1	2	...	25	26	27
Precision	87.3	94.1	...	96.0	**97.2**	96.6
Recall	86.7	93.2	...	95.5	**97.0**	96.3
F_1-measure	86.9	93.6	...	95.7	**97.1**	96.4

Table 3: Cross validation for *temporal* vs. *atemporal* at each iteration. Hybrid Expansion.

Steps	1	2	...	15	16	17
Precision	80.0	75.7	...	95.7	**96.4**	95.6
Recall	80.1	74.3	...	95.1	**96.0**	95.0
F_1-measure	80.0	74.9	...	95.4	**96.2**	95.3

Table 4: Cross validation for *past*, *present* and *future* at each iteration. Hybrid Expansion.

[5]Different configurations as well as different similarity metrics have been tested but these experiments are out-of-the-scope of this paper.

Representation	Uni.+SW	Uni.+SW+Wn	Uni.+SW+TWnL	Uni.+SW+TWnP	Uni.+SW+TWnH
Precision	85.8	85.6	87.8	**89.8**	89.5
Recall	85.7	85.3	87.8	**89.5**	89.4
F_1-measure	85.8	85.4	87.8	**89.6**	89.4

Table 5: Evaluation results for sentence classification with different TempoWordNets. Balanced corpus: 346 sentences for *past*, 346 sentences for *present* and 346 sentences for *future*.

Evaluation: In order to intrinsically evaluate the time-tagged WordNets (TempoWordNets), we first performed an inter-annotation process over samples of 50 automatically time-tagged WordNet synsets. In particular, three different annotators were presented with *temporal* synsets and their respective glosses, and had to decide upon their correct classification (*temporal* vs. *atemporal*). The results of the multirater agreement evaluation are presented in Table 6. In particular, we processed the free-marginal multirater kappa values (Randolph, 2005) and the fixed-marginal multirater kappa (Siegel and Castellan, 1988) as no bias is present in the data. Overall figures assess moderate agreement for the three TempoWordNets: TWnL for the lexico-semantic expansion, TWnP for the probabilistic expansion and TWnH for the hybrid expansion.

Metric	TWnL	TWnP	TWnH
Fixed-marginal κ	0.5073	0.5199	0.4197
Free-marginal κ	0.5199	0.5199	0.4399

Table 6: Inter-annotator agreement.

These results evidence the difficulty of the task for humans as they do not agree on a great deal of decisions. This is particularly due to the fact that the temporal dimension of synsets is judged upon their glosses and not directly on their inherent concept. For example, "dinosaur" can be classified as *temporal* or *atemporal* as its gloss *any of numerous extinct terrestrial reptiles of the Mesozoic era* allows both interpretations.

So, we performed a new experiment based on those examples where human annotator agreement was 100%. From this dataset, we performed an inter-annotator agreement process with four annotators (three human annotators plus the classifier). The underlying idea is to understand to what extent the built TempoWordNets comply with the "easy" cases. Results are illustrated in Table 7 and clearly show the enhanced intrinsic quality of the hybrid expansion strategy with an almost adequate agreement for the free-marginal κ.

Metric	TWnL	TWnP	TWnH
Fixed-marginal κ	0.4133	0.4767	0.5655
Free-marginal κ	0.4242	0.5161	0.6896

Table 7: Inter-annotation for "easy" cases.

5 Sentence Temporal Classification

In order to evaluate TempoWordNets, we proposed to test their capability to enhance the external task of sentence temporal classification. For that purpose, we used the corpus developed by (Dias et al., 2014), which contains 1455 sentences distributed as follows: 724 for *past*, 385 for *present* and 346 for *future*. Different sentence representations have been used. First, we proposed to represent each sentence with the classical vector space model using the tf.idf weighting scheme for unigrams without stop-words removal (Uni.+SW). Then, we proposed a semantic vector space representation where each sentence is augmented with the synonyms of any temporal word contained in it. In particular, we proposed that the words were matched directly from the WordNet time subtree (Uni.+SW+Wn) or from TempoWordNet (Uni.+SW+TWnL, Uni.+SW+TWnP and Uni.+SW+TWnH) and weighted with tf.idf. The results of our experiments are reported in Table 5. The results evidence that the WordNet time subtree does not embody enough time-related information and the process of automatically time-tagging WordNet can improve the task of sentence temporal classification, especially with the probabilistic or the hybrid expansion.

6 Conclusion

In this paper, we proposed the first steps towards the automatic construction of temporal ontologies. In particular, we presented and evaluated different propagation strategies to time tag WordNet giving rise to different TempoWordNets. First results are promising and we deeply believe that such a resource can be important for time related applications both in NLP and IR. All resources can be found at http://tempowordnet.greyc.fr.

References

O. Alonso, R. Baeza-Yates, and M. Gertz. 2007. Exploratory search using timelines. In *Proceedings of the ACM SIGCHI Workshop on Exploratory Search and HCI*.

O. Alonso, M. Gertz, and R. Baeza-Yates. 2009. Clustering and exploring search results using timeline constructions. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM)*, pages 97–106. ACM.

O. Alonso, J. Strötgen, R. Baeza-Yates, and M. Gertz. 2011. Temporal information retrieval: Challenges and opportunities. In *Proceedings of the 1st International Temporal Web Analytics Workshop (TWAW)*, pages 1–8.

A. Anand, S. Bedathur, K. Berberich, and R. Schenkel. 2012. Index maintenance for time-travel text search. In *Proceedings of the 35th International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 235–244.

Ricardo Baeza-Yates. 2005. Searching the future. In *Proceedings of the ACM SIGIR Workshop on Mathematical/Formal Methods in Information Retrieval*, pages 1–6.

L. Derczynski and R. Gaizauskas. 2012. A corpus-based study of temporal signals. *arXiv:1203.5066*.

G. Dias, J.G. Moreno, A. Jatowt, and R. Campos. 2012. Temporal web image retrieval. In *Proceedings of the 19th Edition of the International Symposium on String Processing and Information Retrieval (SPIRE)*, pages 199–204. Springer.

G. Dias, Md. Hasanuzzaman, S. Ferrari, and Y. Mathet. 2014. Tempowordnet for sentence time tagging. In *Proceedings of the 4th ACM Temporal Web Analytics Workshop (TEMPWEB)*.

A. Esuli and F. Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *Proceedings of the 5th Conference on Language Resources and Evaluation (LREC)*, pages 417–422.

C. Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. Bradford Books.

A. Jatowt, C.-M. Au Yeung, and K. Tanaka. 2013. Estimating document focus time. In *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management (CIKM)*, pages 2273–2278.

N. Kanhabua, R. Blanco, and M. Matthews. 2011. Ranking related news predictions. In *Proceedings of the 34th International ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 755–764.

A. Kulkarni, J. Teevan, K.M. Svore, and S. Dumais. 2011. Understanding temporal query dynamics. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 167–176.

C. Leacock, G.A. Miller, and M. Chodorow. 1998. Using corpus statistics and wordnet relations for sense identification. *Computational Linguisics*, 24(1):147–165.

I. Mani, J. Pustejovsky, and R. Gaizauskas. 2005. *The language of time: a reader*, volume 126. Oxford University Press.

M.J. Metzger. 2007. Making sense of credibility on the web: Models for evaluating online information and recommendations for future research. *Journal of the American Society for Information Science and Technology*, 58(13):2078–2091.

D. Metzler, R. Jones, F. Peng, and R. Zhang. 2009. Improving search relevance for implicitly temporal queries. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, pages 700–701.

G.A. Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

M. Moens and M. Steedman. 1987. Temporal ontology in natural language. In *Proceedings of the 25th Annual Meeting on Association for Computational Linguistics (ACL)*, pages 1–7.

J. Pustejovsky and M. Verhagen. 2009. Semeval-2010 task 13: evaluating events, time expressions, and temporal relations (tempeval-2). In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 112–116.

J. Pustejovsky, B. Ingria, R. Sauri, J. Castano, J. Littman, R. Gaizauskas, A. Setzer, G. Katz, and I. Mani. 2005. The specification language timeml. *The language of time: A reader*, pages 545–557.

K. Radinsky and E. Horvitz. 2013. Mining the web to predict future events. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM)*, pages 255–264.

J.J. Randolph. 2005. Free-marginal multirater kappa (multirater κfree): an alternative to fleiss' fixed-marginal multirater kappa. *Joensuu Learning and Instruction Symposium*.

N. Siegel and J.N. Castellan. 1988. *Nonparametric Statistics for the Social Sciences*. Mcgraw-hill edition.

J. Strötgen and M. Gertz. 2013. Multilingual and cross-domain temporal tagging. *Language Resources and Evaluation (LRE)*, 47(2):269–298.

N. UzZaman and J.F. Allen. 2010. Trips and trios system for tempeval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 276–283.

N. UzZaman, H. Llorens, L. Derczynski, M. Verhagen, J. Allen, and J. Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*.

M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, G. Katz, and J. Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 75–80.

M. Verhagen, R. Gaizauskas, F. Schilder, M. Hepple, J. Moszkowicz, and J. Pustejovsky. 2009. The tempeval challenge: Identifying temporal relations in text. *Language Resources and Evaluation (LRE)*, 43(2):161–179.

Chinese Open Relation Extraction for Knowledge Acquisition

Yuen-Hsien Tseng[1], Lung-Hao Lee[1,2], Shu-Yen Lin[1], Bo-Shun Liao[1], Mei-Jun Liu[1], Hsin-Hsi Chen[2], Oren Etzioni[3], Anthony Fader[4]

[1]Information Technology Center, National Taiwan Normal University
[2]Dept. of Computer Science and Information Engineering, National Taiwan University
[3]Allen Institute for Artificial Intelligence, Seattle, WA
[4]Dept. of Computer Science and Engineering, University of Washington
{samtseng, lhlee, sylin, skylock, meijun}@ntnu.edu.tw,
hhchen@ntu.edu.tw, OrenE@allenai.org, afader@cs.washington.edu

Abstract

This study presents the Chinese Open Relation Extraction (CORE) system that is able to extract entity-relation triples from Chinese free texts based on a series of NLP techniques, *i.e.,* word segmentation, POS tagging, syntactic parsing, and extraction rules. We employ the proposed CORE techniques to extract more than 13 million entity-relations for an open domain question answering application. To our best knowledge, CORE is the first Chinese Open IE system for knowledge acquisition.

1 Introduction

Traditional Information Extraction (IE) involves human intervention of handcrafted rules or tagged examples as the input for machine learning to recognize the assertion of a particular relationship between two entities in texts (Riloff, 1996; Soderland, 1999). Although machine learning helps enumerate potential relation patterns for extraction, this approach is often limited to extracting the relation sets that are predefined. In addition, traditional IE has focused on satisfying pre-specified requests from small homogeneous corpora, leaving the question open whether it can scale up to massive and heterogeneous corpora such as the Web (Banko and Etzioni, 2008; Etzioni et al., 2008, 2011).

Open IE, a new domain-independent knowledge discovery paradigm that extracts a diverse set of relations without requiring any relation-specific human inputs and a pre-specified vocabulary, is especially suited to massive text corpora, where target relations are unknown in advance. Several Open IE systems, such as TextRunner (Banko et al., 2007), WOE (Wu and Weld, 2010), ReVerb (Fader et al., 2011), and OLLIE (Mausam et al., 2012) achieve promising performance in open relation extraction on English sentences. However, application of these systems poses challenges to those languages that are very different from English, such as Chinese, as grammatical functions in English and Chinese are realized in markedly different ways. It is not sure whether those techniques for English still work for Chinese. This issue motivates us to extend the state-of-the-art Open IE systems to extract relations from Chinese texts.

The relatively rich morpho-syntactic marking system of English (e.g., verbal inflection, nominal case, clausal markers) makes the syntactic roles of many words detectable from their surface forms. A tensed verb in English, for example, generally indicates its main verb status of a clause. The pinning down of the main verb in a Chinese clause, on the other hand, must rely on other linguistic cues such as word context due to the lack of tense markers. In contrast to the syntax-oriented English language, Chinese is discourse-oriented and rich in ellipsis – meaning is often construable in the absence of explicit linguistic devices such that many obligatory grammatical categories (e.g., pronouns and BE verbs) can be elided in Chinese. For example, the three Chinese sentences “蘋果營養豐富” (‘Apples nutritious’), “蘋果是營養豐富的” (‘Apples are nutritious’), and “蘋果富含營養”

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 12–16,
Gothenburg, Sweden, April 26-30 2014.

('Apples are rich in nutrition') are semantically synonymous sentences, but the first one, which lacks an overt verb, is used far more often than the other two. Presumably, an adequate multilingual IE system must take into account those intrinsic differences between languages.

This paper introduces the Chinese Open Relation Extraction (CORE) system, which utilizes a series of NLP techniques to extract relations embedded in Chinese sentences. Given a Chinese text as the input, CORE employs word segmentation, part-of-speech (POS) tagging, and syntactic parsing, to automatically annotate the Chinese sentences. Based on this rich information, the input sentences are chunked and the entity-relation triples are extracted. Our evaluation shows the effectiveness of CORE, and its deficiency as well.

2 Related Work

TextRunner (Banko et al., 2007) was the first Open IE system, which trains a Naïve Bayes classifier with POS and NP-chunk features to extract relationships between entities. The subsequent work showed that employing the classifiers capable of modeling the sequential information inherited in the texts, like linear-chain CRF (Banko and Etzioni, 2008) and Markov Logic Network (Zhu et al., 2009), can result in better extraction performance. The WOE system (Wu and Weld, 2010) adopted Wikipedia as the training source for their extractor. Experimental results indicated that parsed dependency features lead to further improvements over TextRunner.

ReVerb (Fader et al., 2011) introduced another approach by identifying first a verb-centered relational phrase that satisfies their pre-defined syntactic and lexical constraints, and then split the input sentence into an Argument-Verb-Argument triple. This approach involves only POS tagging for English and "regular expression"-like matching. As such, it is suitable for large corpora, and likely to be applicable to Chinese.

For multilingual open IE, Gamallo et al. (2012) adopts a rule-based dependency parser to extract relations represented in English, Spanish, Portuguese, and Galician. For each parsed sentence, they separate each verbal clause and then identify each one's verb participants, including their functions: subject, direct object, attribute, and prepositional complements. A set of rules is then applied on the clause constituents to extract the target triples. For Chinese open IE, we adopt a similar general approach. The main differences are the processing steps specific to Chinese language.

3 Chinese Open Relation Extraction

This section describes the components of CORE. Not requiring any predefined vocabulary, CORE's sole input is a Chinese corpus and its output is an extracted set of relational tuples. The system consists of three key modules, i.e., word segmentation and POS tagging, syntactic parsing, and entity-relation triple extraction, which are introduced as follows:

Chinese is generally written without word boundaries. As a result, prior to the implementation of most NLP tasks, texts must undergo automatic word segmentation. Automatic Chinese word segmenters are generally trained by an input lexicon and probability models. However, it usually suffers from the unknown word (i.e., the out-of-vocabulary, or OOV) problem. In CORE, a corpus-based learning method to merge the unknown words is adopted to tackle the OOV problem (Chen and Ma, 2002). This is followed by a reliable and cost-effective POS-tagging method to label the segmented words with part-of-speeches (Tsai and Chen, 2004). Take the Chinese sentence "愛迪生發明了燈泡" ('Edison invented the light bulb') for instance. It was segmented and tagged as follows: 愛迪生/Nb 發明/VC 了/Di 燈泡/Na. Among these words, the translation of a foreign proper name "愛迪生" ('Edison') is not likely to be included in a lexicon and therefore is extracted by the unknown word detection method. In this case,

the special POS tag 'Di' is a tag to represent a verb's tense when its character "了" follows immediately after its precedent verb. The complete set of part-of-speech tags is defined in the technical report (CKIP, 1993). In the above sentence, "了" could represent a complete different meaning if it is associate with other character, such as "了解" meaning "understand". Therefore, "愛迪生發明了解藥" ('Edison invented a cure') would be segmented incorrectly once "了" is associated with its following character, instead of its precedent word.

We adopt CKIP, the best-performing parser in the bakeoff of SIGHAN 2012 (Tseng et al., 2012), to do syntactic structure analysis. The CKIP solution re-estimates the context-dependent probability for Chinese parsing and improves the performance of probabilistic context-free grammar (Hsieh et al., 2012). For the example sentence above, '愛迪生/Nb' and '燈泡/Na' were annotated as two nominal phrases (i.e., 'NP'), and '發明/VC 了/Di' was annotated as a verbal phrase (i.e., 'VP').

CKIP parser also adopts dependency decision-making and example-based approaches to label the semantic role "Head", showing the status of a word or a phrase as the pivotal constituent of a sentence (You and Chen, 2004). CORE adopts the *head-driven principle* to identify the main relation in a given sentence (Huang et al., 2000). Firstly, a relation is defined by both the "Head"-labeled verb and the other words in the syntactic chunk headed by the verb. Secondly, the noun phrases preceding/preceded by the relational chunk are regarded as the candidates of the head's arguments. Finally, the entity-relation triple is identified in the form of (entity1, relation, entity2). Regarding the example sentence described above, the triple (愛迪生/Edison, 發明了/invented, 燈泡/light bulb) is extracted by this approach.

Figure 1 shows the parsed tree of a Chinese sentence for the relation extraction by CORE. The Chinese sentence "白宮預算委員會的民主黨星期一發佈報告" ('Democrats on the House Budget Committee released a report on Monday') is the manual translation of one of the English sentences evaluated by ReVerb (Fader et al., 2011). The first step of CORE involves word-segmentation and POS-tagging, thus returning eight word/POS pairs: 白宮/Nc, 預算/Na, 委員會/Nc, 的/DE, 民主黨/Nb, 星期一/Nd, 發佈/VE, 報告/Na. Next, "星期一/Nd 發佈/VE" is identified as the verbal phrase that heads the sentence. This verbal phrase is regarded as the center of a potential relation. The two noun phrases before and after the verbal phrase, *i.e.*, the NP "白宮 預算 委員會 的 民主黨" and NP "報告" are regarded as the entities that complete the relation. A potential entity-relation-entity triple (*i.e.*, 白宮預算委員會的民主黨 / 星期一發佈 / 報告, 'Democrats on the House Budget Committee / on Monday released / a report') is extracted accordingly. This triple is chunked from its original sentence fully automatically. Finally, a filtering process, which retains "Head"-labeled words only, can be applied to strain out from each component of this triple the most prominent word: "民主黨 / 發佈 / 報告" ('Democrats / released / report').

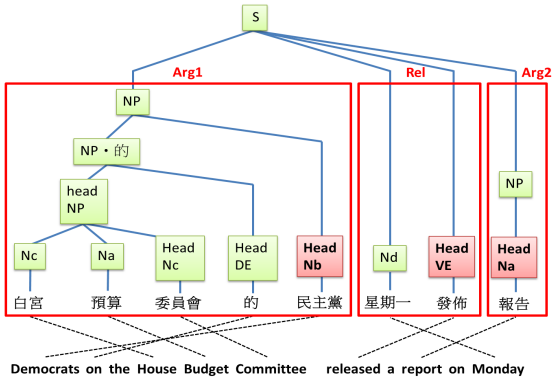


Figure 1: The parsed tree of a Chinese sentence.

4 Experiments and Evaluation

We adopted the same test set released by ReVerb for performance evaluation. The test set consists of 500 English sentences randomly sampled from the Web and were annotated using a pooling method. To obtain "gold standard" relation triples in Chinese, the 500 test sentences were manually translated from English to Chinese by a

trained native Chinese speaker and verified by another. Additionally, two other native Chinese speakers annotated the relation triples for each Chinese sentence. In total, 716 Chinese entity-relation triples with an agreement score of 0.79 between the two annotators were obtained and regarded as gold standard.

Performance evaluation of CORE was conducted based on: 1) exact match; and 2) relation-only match. For exact match, each component of the extracted triple must be identical with the gold standard. For relation-only match, the extracted triple is regarded as a correct case if an extracted relation agreed with the relation of the gold standard.

Without another Chinese Open IE system for performance comparison, we compared CORE with a modification of ReVerb system capable of handling Chinese sentences. The modification of ReVerb's verb-driven regular expression matching was kept to a minimum to deal with language-specific processing. As such, ReVerb remains mostly the same as its English counterpart so that a bilingual (Chinese/English) Open IE system can be easily implemented.

Table 1 shows the experimental results. Our CORE system obviously performs better than ReVerb when recall is considered for both exact and relation-only match. The results suggest that utilizing more sophisticated NLP techniques is effective to extract relations without any specific human intervention. In addition, there is a slight decrease in the precision of exact match for CORE. This reveals that ReVerb's original syntactic and lexical constraints are also useful to identify the arguments and their relationship precisely. In summary, CORE achieved relatively promising F1 scores. These results imply that CORE method is more suitable for Chinese open relation extraction.

Chinese Open IE		**Precision**	**Recall**	**F1**
Exact Match	ReVerb	**0.5820**	0.0987	0.1688
	CORE	0.5579	**0.3291**	**0.4140**
Relation Only	ReVerb	0.8361	0.1425	0.2435
	CORE	**0.8463**	**0.5000**	**0.6286**

Table 1: Performance evaluation on Chinese Open IE.

We also analyzed the errors made by the CORE model. Almost all the errors resulted from incorrect parsing. Enhancing the parsing effectiveness is most likely to improve the performance of CORE. The relatively low recall rate also indicates that CORE misses many types of relation expression. Ellipsis and flexibility in Chinese syntax are so difficult not only to fail the parser, but also the extraction attempts to bypass the parsing errors.

To demonstrate the applicability of CORE, we implement a Chinese Question-Answering (QA) system based on two million news articles from 2002 to 2009 published by the United Daily News Group (udn.com/NEWS). CORE extracted more than 13 million unique entity-relation triples from this corpus. These extracted relations are useful for knowledge acquisition. Take the question "什麼源自於中國？" ('What is originated from China?') as an example, the relation is automatically identified as "源" ('originate') that heads the following entity "中國" ('China'). Our open QA system then searched the triples and returned the first entity as the answers. In addition to the obvious answer "中醫" ('Chinese medicine'), which is usually considered as common-sense knowledge, we also obtained those that are less known, such as the traditional Japanese food "納豆" ('natto') and the musical instrument "手風琴" ('accordion').

5 Conclusions

This work demonstrates the feasibility of extracting relations from Chinese corpus without the input of any predefined vocabulary to IE systems. This work is the first to explore Chinese open relation extraction to our best knowledge.

Acknowledgments

This research was partially supported by National Science Council, Taiwan under grant NSC102-2221-E-002-103-MY3, and the "Aim for the Top University Project" of National Taiwan Normal University, sponsored by the Ministry of Education, Taiwan.

References

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. *Proceedings of EMNLP'11*, pages 1535-1545.

Chu-Ren Huang, Feng-Yi Chen, Keh-Jiann Chen, Zhao-Ming Gao, and Kuang-Yu Chen. 2000. Sinina Treebank: design criteria, annotation guidelines, and on-line interface. *Proceedings of SIGHAN'00*, pages 29-37.

Chinese Knowledge Information Processing (CKIP) Group. 1993. Categorical analysis of Chinese. *ACLCLP Technical Report* # 93-05, Academia Sinica.

Fei Wu and Daniel S. Weld. 2010. Open information extraction using Wikipedia. *Proceedings of ACL'10*, pages 118-127.

Jia-Ming You, and Keh-Jiann Chen. 2004. Automatic semantic role assignment for a tree structure. In *Proceedings of SIGHAN'04*, pages 1-8.

Jun Zhu, Zaiqing Nie, Xiaojiang Lium Bo Zhang, and Ji-Rong Wen. 2009. StatSnowball: a statistical approach to extracting entity relationships. In *Proceedings of WWW'09*, pages 101-110.

Keh-Jiann Chen and Wei-Yun Ma. 2002. Unknown word extraction for Chinese documents. In *Proceedings of COLING'02*, pages 169-175.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. *Proceedings of IJCAI'07*, pages 2670-2676.

Michele Banko, and Oren Etzioni. 2008. The tradeoffs between open and traditional relation extraction. *Proceedings of ACL'08*, pages 28-26.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam. 2011. Open information extraction: the second generation. In *Proceedings of IJCAI'11*, pages 3-10.

Oren Etzioni, Michele Banko, Stephen Soderland, and Daniel S. Weld. 2008. Open information extraction from the web. *Communications of the ACM*, 51(12):68-74.

Pablo Gamallo, Marcos Garcia, and Santiago Fernández-Lanza. 2012. Dependency-based open information extraction. In *Proceedings of ROBUS-UNSUP'12*, pages 10-18.

Elleen Riloff. 1996. Automatically constructing extraction patterns from untagged text. In *Proceedings of AAAI'96*, pages 1044-1049.

Stephen Soderland. 1999. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1-3):233-272.

Yu-Ming Hsieh, Ming-Hong Bai, Jason S. Chang, and Keh-Jiann Chen. 2012. Improving PCFG Chinese Parsing with Context-Dependent Probability Re-estimation. *Proceedings of CLP'12*, pages 216-221.

Yu-Fang Tsai, and Keh-Jiann Chen. 2004. Reliable and cost-effective pos-tagging. *International Journal of Computational Linguistics and Chinese Language Processing*, 9(1):83-96.

Yuen-Hsien Tseng, Lung-Hao Lee, and Liang-Chih Yu 2012. Traditional Chinese parsing evaluation at SIGHAN Bake-offs 2012. *Proceedings of* CLP'12, pages 199-205.

Temporal Text Ranking and Automatic Dating of Texts

Vlad Niculae[1], Marcos Zampieri[2], Liviu P. Dinu[3], Alina Maria Ciobanu[3]
Max Planck Institute for Software Systems, Germany[1]
Saarland University, Germany[2]
Center for Computational Linguistics, University of Bucharest, Romania[3]
vniculae@mpi-sws.org, marcos.zampieri@uni-saarland.de,
ldinu@fmi.unibuc.ro, alina.ciobanu@my.fmi.unibuc.ro

Abstract

This paper presents a novel approach to the task of temporal text classification combining text ranking and probability for the automatic dating of historical texts. The method was applied to three historical corpora: an English, a Portuguese and a Romanian corpus. It obtained performance ranging from 83% to 93% accuracy, using a fully automated approach with very basic features.

1 Introduction

Temporal text classification is an underexplored problem in NLP, which has been tackled as a multi-class problem, with classes defined as time intervals such as months, years, decades or centuries. This approach has the drawback of having to arbitrarily delimit the intervals, and often leads to a model that is not informative for texts written within such a window. If the predefined window is too large, the output is not useful for most systems; if the window is too small, learning is impractical because of the large number of classes. Particularly for the problem of historical datasets (as the one we propose here), learning a year-level classifier would not work, because each class would be represented by a single document.

Our paper explores a solution to this drawback by using a *ranking* approach. Ranking amounts to ordering a set of inputs with respect to some measure. For example, a search engine ranks returned documents by relevance. We use a formalization of ranking that comes from *ordinal regression*, the class of problems where samples belong to inherently ordered classes.

This study is of interest to scholars who deal with text classification and NLP in general; historical linguists and philologists who investigate language change; and finally scholars in the digital humanities who often deal with historical manuscripts and might take advantage of temporal text classification applications in their research.

2 Related Work

Modelling temporal information in text is a relevant task for a number of NLP tasks. For example, in Information Retrieval (IR) research has been concentrated on investigating time-sensitivity document ranking (Dakka and Gravana, 2010). Even so, as stated before, temporal text classification methods were not substantially explored as other text classification tasks.

One of the first studies to model temporal information for the automatic dating of documents is the work of de Jong et al. (2005). In these experiments, authors used unigram language models to classify Dutch texts spanning from January 1999 to February 2005 using normalised log-likelihood ratio (NLLR) (Kraaij, 2004). As to the features used, a number of approaches proposed to automatic date take into account lexical features (Dalli and Wilks, 2006; Abe and Tsumoto, 2010; Kumar et al., 2011) and a few use external linguistic knowledge (Kanhabua and Nørvåg, 2009).

A couple of approaches try to classify texts not only regarding the time span in which the texts were written, but also their geographical location such as (Mokhov, 2010) for French and, more recently, (Trieschnigg et al., 2012) for Dutch. At the word level, two studies aim to model and understand how word usage and meaning change over time (Wijaya and Yeniterzi, 2011), (Mihalcea and Nastase, 2012).

The most recent studies in temporal text classification to our knowledge are (Ciobanu et al., 2013) for Romanian using lexical features and (Štajner and Zampieri, 2013) for Portuguese using stylistic and readability features.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 17–21,
Gothenburg, Sweden, April 26-30 2014.

3 Methods

3.1 Corpora

To evaluate the method proposed here we used three historical corpora. An English historical corpus entitled Corpus of Late Modern English Texts (CLMET)[1] (de Smet, 2005), a Portuguese historical corpus entitled Colonia[2] (Zampieri and Becker, 2013) and a Romanian historical corpus (Ciobanu et al., 2013).

CLMET is a collection of English texts derived from the Project Gutenberg and from the Oxford Text Archive. It contains around 10 million tokens, divided over three sub-periods of 70 years. The corpus is available for download as raw text or annotated with POS annotation.

For Portuguese, the aforementioned Colonia (Zampieri and Becker, 2013) is a diachronic collection containing a total of 5.1 million tokens and 100 texts ranging from the 16th to the early 20th century. The texts in Colonia are balanced between European and Brazilian Portuguese (it contains 52 Brazilian texts and 48 European texts) and the corpus is annotated with lemma and POS information. According to the authors, some texts presented edited orthography prior to their compilation but systematic spelling normalisation was not carried out.

The Romanian corpus was compiled to portrait different stages in the evolution of the Romanian language, from the 16th to the 20th century in a total of 26 complete texts. The methodology behind corpus compilation and the date assignment are described in (Ciobanu et al., 2013).

3.2 Temporal classification as ranking

We propose a temporal model that learns a linear function $g(x) = w \cdot x$ to preserve the temporal ordering of the texts, i.e. if document[3] x_i predates document x_j, which we will henceforth denote as $x_i \prec x_j$, then $g(x_i) < g(x_j)$. Such a problem is often called *ranking* or *learning to rank*. When the goal is to recover contiguous intervals that correspond to ordered classes, the problem is known as *ordinal regression*.

We use a pairwise approach to ranking that reduces the problem to binary classification using a linear model. The method is to convert a dataset of the form $\mathcal{D} = \{(x, y) : x \in \mathbb{R}^d, y \in \mathcal{Y}\}$ into a pairwise dataset:

$$\mathcal{D}_p = \{((x_i, x_j), \mathbf{I}[y_i < y_j]) : (x_i, y_i), (x_j, y_j) \in \mathcal{D}\}$$

Since the ordinal classes only induce a partial ordering, as elements from the same class are not comparable, $\mathcal{D}_p$ will only consist of the comparable pairs.

The problem can be turned into a linear classification problem by noting that:

$$w \cdot x_i < w \cdot x_j \iff w \cdot (x_i - x_j) < 0$$

In order to obtain probability values for the ordering, we use logistic regression as the linear model. It therefore holds that:

$$\mathbf{P}(x_i \prec x_j; w) = \frac{1}{1 + \exp(-w \cdot (x_i - x_j))}$$

While logistic regression usually fits an intercept term, in our case, because the samples consist of differences of points, the model operates in an affine space and therefore gains an extra effective degree of freedom. The intercept is therefore not needed.

The relationship between pairwise ranking and predicting the class from an ordered set $\{r_1, ...r_k\}$ is given by assigning to a document x the class r_i such that

$$\theta(r_{i-1}) \leq g(x) < \theta(r_i) \quad (1)$$

where θ is an increasing function that does not need to be linear. (Pedregosa et al., 2012), who used the pairwise approach to ordinal regression on neuroimaging prediction tasks, showed using artificial data that θ can be accurately recovered using non-parametric regression. In this work, we use a parametric estimation of θ that can be used in a probabilistic interpretation to identify the most likely period when a text was written, as described in section 3.3.

3.3 Probabilistic dating of uncertain texts

The ranking model described in the previous section learns a direction along which the temporal order of texts is preserved as much as possible. This direction is connected to the chronological axis through the θ function. For the years t for

[1] https://perswww.kuleuven.be/~u0044428/clmet

[2] http://corporavm.uni-koeln.de/colonia/

[3] For brevity, we use x_i to denote both the document itself and its representation as a feature vector.

which we have an unique attested document x_t, we have that

$$x \prec x_t \iff g(x) < g(x_t) < \theta(t)$$

This can be explained by seeing that equation 2 gives $\theta(t)$ as an upper bound for the projections of all texts written in year t, and by transitivity for all previous texts as well.

Assuming we can estimate the function θ with another function $\hat{\theta}$, the cumulative densitiy function of the distribution of the time when an unseen document was written can be expressed.

$$P(x \prec t) \approx \frac{1}{1 + \exp(w \cdot x - \hat{\theta}(t))} \qquad (2)$$

Setting the probability to $\frac{1}{2}$ provides a point estimate of the time when x was written, and confidence intervals can be found by setting it to p and $1 - p$.

3.4 Features

Our ranking and estimation model can work with any kind of numerical features. For simplicity we used lexical and naive morphological features, pruned using χ^2 feature selection with tunable granularity.

The lexical features are occurrence counts of all words that appear in at least p_{lex} documents. The morphological features are counts of character n-grams of length up to w_{mph} in final positions of words, filtered to occur in at least n_{mph} documents.

Subsequently, a non-linear transformation ϕ is optionally applied to the numerical features. This is one of $\phi_{\text{sqrt}}(z) = \sqrt{z}$, $\phi_{\log}(z) = \log(z)$ or $\phi_{\text{id}}(z) = z$ (no transformation).

The feature selection step is applied before generating the pairs for classification, in order for the χ^2 scoring to be applicable. The raw target values used are year labels, but to avoid separating almost every document in its own class, we introduce a *granularity* level that transforms the labels into groups of n_{gran} years. For example, if $n_{\text{gran}} = 10$ then the features will be scored according to how well they predict the decade a document was written in. The features in the top p_{fsel} percentile are kept. Finally, C is the regularization parameter of the logistic regression classifier, as defined in *liblinear* (Fan et al., 2008).

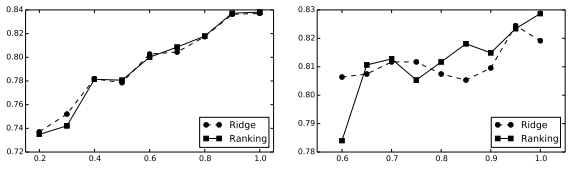


Figure 1: Learning curves for English (top) and Portuguese (bottom). Proportion of training set used versus score.

4 Results

Each corpus is split randomly into training and test sets with equal number of documents. The best feature set is chosen by 3-fold cross-validated random search over a large grid of possible configurations. We use random search to allow for a more efficient exploration of the parameter space, given that some parameters have much less impact to the final score than others.

The evaluation metric we used is the percentage of non-inverted (correctly ordered) pairs, following (Pedregosa et al., 2012).

We compare the pairwise logistic approach to a ridge regression on the same feature set, and two multiclass SVMs, at century and decade level. While the results are comparable with a slight advantage in favour of ranking, the pairwise ranking system has several advantages. On the one hand, it provides the probabilistic interpretation described in section 3.3. On the other hand, the model can naturally handle noisy, uncertain or wide-range labels, because annotating whether a text was written before another can be done even when the texts do not correspond to punctual moments in time. While we do not exploit this advantage, it can lead to more robust models of temporal evolution. The learning curves in Figure 1 further show that the pairwise approach can better exploit more data and nonlinearity.

The implementation is based on the *scikit-learn* machine learning library for Python (Pedregosa et al., 2011) with logistic regression solver from (Fan et al., 2008). The source code will be available.

4.1 Uncertain texts

We present an example of using the method from Section 3.3 to estimate the date of uncertain, held-out texts of historical interest. Figure 2 shows the process used for estimating θ as a linear, and in the case of Portuguese, quadratic function. The

	size	p_{lex}	n_{mph}	w_{mph}	ϕ	n_{gran}	p_{fsel}	C	score	ridge	century	decade	MAE
en	293	0.9	0	3	$\phi_{\log}$	100	0.15	2^{9}	0.838	0.837	0.751	0.813	22.8
pt	87	0.9	25	4	ϕ_{sqrt}	5	0.25	2^{-5}	0.829	0.819	0.712	0.620	58.7
ro	42	0.8	0	4	$\phi_{\log}$	5	0.10	2^{28}	0.929	0.924	0.855	0.792	28.8

Table 1: Test results of the system on the three datasets. The score is the proportion of pairs of documents ranked correctly. The column *ridge* is a linear regression model used for ranking, while *century* and *decade* are linear SVMs used to predict the century and the decade of each text, but scored as pairwise ranking, for comparability. Chance level is 0.5. MAE is the mean absolute error in years. The hyperparameters are described in section 3.4.

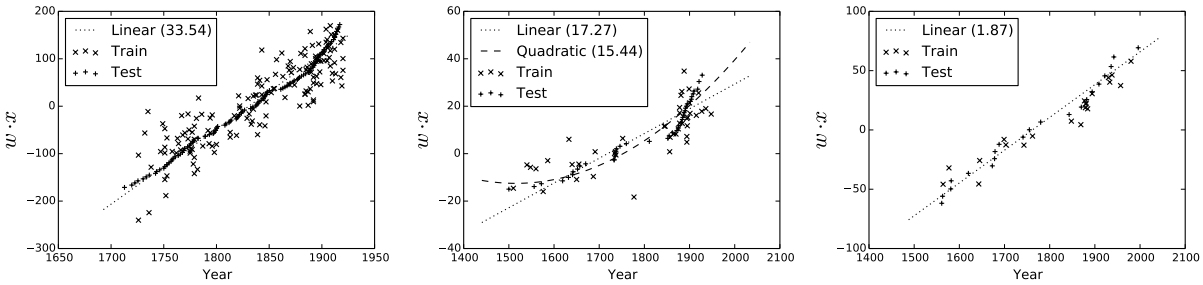


Figure 2: Estimating the function θ that defines the relationship between years and projections of documents to the direction of the model, for English, Portuguese and Romanian (left to right). In parantheses, the normalized residual of the least squares fit is reported on the test set.

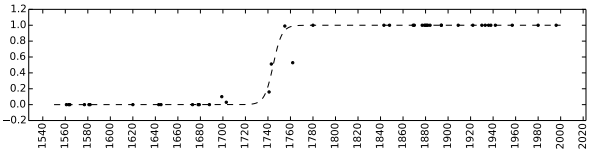


Figure 3: Visualisation of the probability estimation for the dating of C. Cantacuzino's *Istoria Țării Rumânești*. The horizontal axis is the time, the points are known texts with a height equal to the probability predicted by the classifier. The dashed line is the estimated probability from Equation 2.

estimation is refit on all certain documents prior to plugging into the probability estimation.

The document we use to demonstrate the process is Romanian nobleman and historian Constantin Cantacuzino's *Istoria Țării Rumânești*. The work is believed to be written in 1716, the year of the author's death, and published in several editions over a century later (Stahl, 2001). This is an example of the system being reasonably close to the hypothesis, thus providing linguistic support to it. Our system gives an estimated dating of 1744.7 with a 90% confidence interval of $1736.2 - 1753.2$. As publications were significantly later, the lexical pull towards the end of 18th century that can be observed in Figure 3 could be driven by possible editing of the original text.

5 Conclusion

We propose a ranking approach to temporal modelling of historical texts. We show how the model can be used to produce reasonable probabilistic estimates of the linguistic age of a text, using a very basic, fully-automatic feature extraction step and no linguistic or historical knowledge injected, apart from the labels, which are possibly noisy.

Label noise can be atenuated by replacing uncertain dates with intervals that are more certain, and only generating training pairs out of non-overlapping intervals. This can lead to a more robust model and can use more data than would be possible with a regression or classification approach. The problem of potential edits that a text has suffered still remains open.

Finally, better engineered and linguistically-motivated features, such as syntactic, morphological or phonetic patterns that are known or believed to mark epochs in the evolution of a language, can be plugged in with no change to the fundamental method.

References

H. Abe and S. Tsumoto. 2010. Text categorization with considering temporal patterns of term usages. In *Proceedings of ICDM Workshops*, pages 800–807. IEEE.

A. Ciobanu, A. Dinu, L. Dinu, V. Niculae, and O. Sulea. 2013. Temporal text classification for romanian novels set in the past. In *Proceedings of RANLP2013*, Hissar, Bulgaria.

W. Dakka and C. Gravana. 2010. Answering general time-sensitive queries. *IEEE Transactions on Knowledge and Data Engineering*.

A. Dalli and Y. Wilks. 2006. Automatic dating of documents and temporal text classification. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 17–22, Sidney, Australia.

F. de Jong, H. Rode, and D. Hiemstra. 2005. Temporal language models for the disclosure of historical text. In *Proceedings of AHC 2005 (History and Computing)*.

H. de Smet. 2005. A corpus of late modern english. *ICAME-Journal*.

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

N. Kanhabua and P. Nørvåg. 2009. Using temporal language models for document dating. In *ECML/PKDD*, pages 738–741.

W. Kraaij. 2004. *Variations on language modeling for information retrieval*. Ph.D. thesis, University of Twente.

A. Kumar, M. Lease, and J. Baldridge. 2011. Supervised language modelling for temporal resolution of texts. In *Proceedings of CIKM11 of the 20th ACM international conference on Information and knowledge management*, pages 2069–2072.

R. Mihalcea and V. Nastase. 2012. Word epoch disambiguation: Finding how words change over time. In *Proceedings of ACL*, pages 259–263. Association for Computational Linguistics.

S. Mokhov. 2010. A marf approach to deft2010. In *Proceedings of TALN2010*, Montreal, Canada.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Fabian Pedregosa, Alexandre Gramfort, Gaël Varoquaux, Elodie Cauvet, Christophe Pallier, and Bertrand Thirion. 2012. Learning to rank from medical imaging data. *CoRR*, abs/1207.3598.

H.H. Stahl. 2001. *Gânditori şi curente de istorie socială românească*. Biblioteca Institutului Social Român. Ed. Univ. din Bucureşti.

S. Štajner and M. Zampieri. 2013. Stylistic changes for temporal text classification. In *Proceedings of the 16th International Conference on Text Speech and Dialogue (TSD2013), Lecture Notes in Artificial Intelligence (LNAI)*, pages 519–526, Pilsen, Czech Republic. Springer.

D. Trieschnigg, D. Hiemstra, M. Theune, F. de Jong, and T. Meder. 2012. An exploration of language identification techniques for the dutch folktale database. In *Proceedings of LREC2012*.

D. Wijaya and R. Yeniterzi. 2011. Understanding semantic change of words over centuries. In *Proc. of the Workshop on Detecting and Exploiting Cultural Diversity on the Social Web (DETECT)*.

M. Zampieri and M. Becker. 2013. Colonia: Corpus of historical portuguese. *ZSM Studien, Special Volume on Non-Standard Data Sources in Corpus-Based Research*, 5.

Measuring the Similarity between Automatically Generated Topics

Nikolaos Aletras and **Mark Stevenson**
Department of Computer Science,
University of Sheffield,
Regent Court, 211 Portobello,
Sheffield,
United Kingdom S1 4DP
{n.aletras, m.stevenson}@dcs.shef.ac.uk

Abstract

Previous approaches to the problem of measuring similarity between automatically generated topics have been based on comparison of the topics' word probability distributions. This paper presents alternative approaches, including ones based on distributional semantics and knowledge-based measures, evaluated by comparison with human judgements. The best performing methods provide reliable estimates of topic similarity comparable with human performance and should be used in preference to the word probability distribution measures used previously.

1 Introduction

Topic models (Blei et al., 2010) have proved to be useful for interpreting and organising the contents of large document collections. It seems intuitively plausible that some automatically generated topics will be similar while others are dis-similar. For example, a topic about basketball (*team game james season player nba play knicks coach league)* is more similar to a topic about football (*world cup team soccer africa player south game match goal*) than one about the global finance (*fed financial banks federal reserve bank bernanke rule crisis credit*). Methods for automatically determining the similarity between topics have several potential applications, such as analysis of corpora to determine topics being discussed (Hall et al., 2008) or within topic browsers to decide which topics should be shown together (Chaney and Blei, 2012; Gretarsson et al., 2012; Hinneburg et al., 2012).

Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a popular type of topic model but cannot capture such correlations unless the semantic similarity between topics is measured. Other topic models, such as the Correlated Topic Model (CTM) (Blei and Lafferty, 2006), overcome this limitation and identify correlations between topics.

Approaches to identifying similar topics for a range of tasks have been described in the literature but they have been restricted to using information from the word probability distribution to compare topics and have not been directly evaluated. Word distributions have been compared using a variety of measures such as KL-divergence (Li and McCallum, 2006; Wang et al., 2009; Newman et al., 2009), cosine measure (He et al., 2009; Ramage et al., 2009) and the average Log Odds Ratio (Chaney and Blei, 2012). Kim and Oh (2011) also applied the cosine measure and KL-Divergence which were compared with four other measures: Jaccard's Coefficient, Kendall's τ coefficient, Discount Cumulative Gain and Jensen Shannon Divergence (JSD).

This paper compares a wider range of approaches to measuring topic similarity than previous work. In addition these measures are evaluated directly by comparing them against human judgements.

2 Measuring Topic Similarity

We compare measures based on word probability distributions (Section 2.1), distributional semantic methods (Sections 2.2-2.4), knowledge-based approaches (Section 2.5) and their combination (Section 2.6).

2.1 Topic Word Probability Distribution

We first experimented with measures based on comparison of the topics' word distributions (see Section 1), by applying the JSD, KL-divergence and Cosine approaches and the Log Odds Ratio (Chaney and Blei, 2012).

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 22–27,
Gothenburg, Sweden, April 26-30 2014.

2.2 Topic Model Semantic Space

The semantic space generated by the topic model can be used to represent the topics and the topic words. By definition each topic is a probability distribution over the words in the training corpus. For a corpus with D documents and V words, a topic model learns a relation between words and topics, T, as a $T \times V$ matrix, $\mathbf{W}$, that indicates the probability of each word in each topic. $\mathbf{W}$ is the topic model semantic space and each topic word can be represented as a vector, V_i, with topics as features weighted by the probability of the word in each topic. The similarity between two topics is computed as the average pairwise cosine similarity between their top-10 most probable words (**TS-Cos**).

2.3 Reference Corpus Semantic Space

Topic words can also be represented as vectors in a semantic space constructed from an external source. We adapt the method proposed by Aletras and Stevenson (2013) for measuring topic coherence using distributional semantics[1].

Top-N Features A semantic space is constructed considering only the top n most frequent words in Wikipedia (excluding stop words) as context features. Each topic word is represented as a vector of n features weighted by computing the Pointwise Mutual Information (PMI) (Church and Hanks, 1989) between the topic word and each context feature, $\text{PMI}(w_i, w_j)^\gamma$. γ is a variable for assigning more importance to higher PMI values. In our experiments, we set $\gamma = 3$ and found that the best performance is obtained for $n = 5000$. Similarity between two topics is defined as the average cosine similarity of the topic word vectors (**RCS-Cos-N**).

Topic Word Space Alternatively, we consider only the top-10 topic words from the two topics as context features to generate topic word vectors. Then, topic similarity is computed as the pairwise cosine similarity of the topic word vectors (**RCS-Cos-TWS**).

Word Association Topic similarity can also be computed by applying word association measures directly. Newman et al. (2010) measure topic coherence as the average PMI between the topic words. This approach can be adapted to measure topic similarity by computing the average pairwise PMI between the topic words in two topics (**PMI**).

2.4 Training Corpus Semantic Space

Term-Document Space A matrix $\mathbf{X}$ can be created using the training corpus. Each term (row) represents a topic word vector. Element x_{ij} in $\mathbf{X}$ is the tf.idf of the term i in document j. Topic similarity is computed as the pairwise cosine similarity of the topic word vectors (**TCS-Cos-TD**).

Word Co-occurrence in Training Documents Alternatively, we generate a matrix $\mathbf{Z}$ of co-document frequencies. The matrix $\mathbf{Z}$ consists of V rows and columns representing the V vocabulary words. Element z_{ij} is the log of the number of documents that contains the words i and j normalised by the document frequency, DF, of the word j. Mimno et al. (2011) introduced that metric to measure topic coherence. We adapted it to estimate topic similarity by aggregating the co-document frequency of the words between two topics (**Doc-Co-occ**).

2.5 Knowledge-based Methods

UKB (Agirre et al., 2009) is used to generate a probability distribution over WordNet synsets for each word in the vocabulary V of the topic model using the Personalized PageRank algorithm. The similarity between two topic words is calculated by transforming these distributions into vectors and computing the cosine metric. The similarity between two topics is computed by measuring pairwise similarity between their top-10 topic words and selecting the highest score.

Explicit Semantic Analysis (ESA) proposed by Gabrilovich and Markovitch (2007) transforms the topic keywords into vectors that consist of Wikipedia article titles weighted by their relevance to the keyword. For each topic, the centroid is computed from the keyword vectors. Similarity between topics is computed as the cosine similarity of the ESA centroid vectors.

2.6 Feature Combination Using SVR

We also evaluate the performance of a support vector regression system (**SVR**) (Vapnik, 1998) with a linear kernel using a combination of approaches described above as features[2]. The system is trained and tested using 10-fold cross validation.

[1] Wikipedia is used as a reference corpus to count word co-occurrences and frequencies using a context window of ± 10 words centred on a topic word.

[2] With the exception of JSD, features based on the topics' word probability distributions were not used by SVR since it was found that including them reduced performance.

3 Evaluation

Data We created a data set consisting of pairs of topics generated by two topic models (LDA and CTM) over two document collections using different numbers of topics. The first consists of 47,229 news articles from New York Times (NYT) in the GigaWord corpus and the second contains 50,000 articles from ukWAC (Baroni et al., 2009). Each article is tokenised then stop words and words appearing fewer than five times in the corpora removed. This results in a total of 57,651 unique tokens for the NYT corpus and 72,672 for ukWAC.

LDA Topics are learned by training LDA models over the two corpora using *gensim*[3]. The number of topics is set to $T = 50, 100, 200$ and hyperparameters, α and β, are set to $\frac{1}{T}$. Randomly selecting pairs of topics will result to a data set in which the majority of pairs would not be similar. We overcome that problem by assuming that the JSD between likely relevant pairs will be low while it will be higher for less relevant pairs of topics. We selected 800 pairs of topics. 600 pairs represent topics with similar word distributions (in the top 6 most relevant topics ranked by JSD). The remaining 200 pairs were selected randomly.

CTM is trained using the EM algorithm[4]. The number of topics to learn is set to $T = 50, 100, 200$ and the rest of the settings are set to their default values. The topic graph generated by CTM was used to create all the possible pairs between topics that are connected. This results in a total of 70, 468 and 695 pairs in NYT, and a total of 80, 246 and 258 pairs in ukWAC for the 50, 100 and 200 topics respectively.

Incoherent topics are removed using an approach based on distributional semantics (Aletras and Stevenson, 2013). Each topic is represented using the top 10 words with the highest marginal probability.

Human Judgements of Topic Similarity were obtained using an online crowdsourcing platform, Crowdflower. Annotators were provided with pairs of topics and were asked to judge how similar the topics are by providing a rating on a scale of 0 (completely unrelated) to 5 (identical). The average response for each pair was calculated in order to create the final similarity judgement for use as a gold-standard. The average Inter-Annotator agreement (IAA) across all pairs for all of the collections is in the range of 0.53-0.68. The data set together with gold-standard annotations is freely available[5].

4 Results

Table 1 shows the correlation (Spearman) between the topic similarity metrics described in Section 2 and average human judgements for the LDA and CTM topic pairs. It also shows the performance of a **Word Overlap** baseline which measures the number of terms that two topics have in common normalised by the total number of topic terms.

The correlations obtained using the topics' word probability distributions (Section 2.1), i.e. JSD, KL-divergence and Cos, are comparable with the baseline for all of the topic collections and topic models. The metric proposed by Chaney and Blei (2012) also compares probability distributions and fails to perform well on either data set. These results suggest that these metrics may be sensitive to the high dimensionality of the vocabulary. They also assign high similarity to topics that contain ambiguous words, resulting in low correlations with human judgements.

Performance of the cosine of the word vector (TS-Cos) in the Topic Model Semantic Space (Section 2.2) varies implying that the quality of the latent space generated by LDA and CTM is sensitive to the number of topics.

The similarity metrics that use the reference corpus (Section 2.3) consistently produce good correlations for topic pairs generated using both LDA and CTM. The best overall correlation for a single feature in most cases is obtained using average PMI (in a range of 0.43-0.74). The performance of the distributional semantic metric using the Topic Word Space (RCS-Cos-TWS) is comparable and slightly lower for the top-N features (RCS-Cos-N). This indicates that the reference corpus covers a broader range of semantic subjects than the latent space produced by the topic model.

When the term-document matrix from the training corpus is used as a vector space (Section 2.4) performance is worse than when the reference corpus is used. In addition, using co-document frequency derived from the training corpus does not correlate particularly well with human judgements. These methods are sensitive to the size of the corpus, which may be too small to gener-

[3] http://radimrehurek.com/gensim

[4] http://www.cs.princeton.edu/~blei/ctm-c/index.html

[5] http://staffwww.dcs.shef.ac.uk/people/N.Aletras/resources/topicSim.tar.gz

	Spearman's r											
	LDA						CTM					
	NYT			ukWAC			NYT			ukWAC		
Method	50	100	200	50	100	200	50	100	200	50	100	200
Baseline												
Word Overlap	0.32	0.40	0.51	0.22	0.32	0.41	0.56	0.45	0.49	0.35	0.33	0.53
Topic Word Probability Distribution												
JSD	0.37	0.44	0.53	0.29	0.30	0.34	0.59	0.43	0.49	0.38	0.34	0.60
KL-Divergence	0.29	0.29	0.41	0.20	0.24	0.33	0.54	0.39	0.56	0.31	0.29	0.47
Cos	0.31	0.37	0.59	0.30	0.30	0.36	0.58	0.45	0.52	0.50	0.40	0.58
Chaney and Blei (2012)	0.16	0.26	0.18	0.29	0.21	0.25	0.29	0.40	0.31	-0.23	0.12	0.61
Topic Model Semantic Space												
TS-Cos	0.35	0.41	0.67	0.29	0.35	0.42	0.67	0.51	0.49	0.51	0.42	0.42
Reference Corpus Semantic Space												
RCS-Cos-N	0.37	0.46	0.61	0.35	0.32	0.39	0.60	0.47	0.61	0.57	0.42	0.41
RCS-Cos-TWS	0.40	0.54	0.70	0.38	0.43	0.51	0.63	0.59	0.62	0.60	0.55	0.54
PMI	0.43	0.63	0.74	0.43	0.53	0.64	0.68	0.70	**0.64**	0.58	0.62	0.64
Training Corpus Semantic Space												
TCS-Cos-TD	0.36	0.42	0.67	0.29	0.31	0.40	0.64	0.54	0.58	0.49	0.43	0.43
Doc-Co-occ	0.28	0.29	0.45	0.28	0.22	0.30	0.65	0.36	0.57	0.31	0.26	0.34
Knowledge-based												
UKB	0.25	0.38	0.56	0.22	0.35	0.41	0.52	0.41	0.40	0.41	0.43	0.42
ESA	0.43	0.58	0.71	**0.46**	0.55	0.61	0.69	0.67	**0.64**	**0.70**	0.62	0.61
Feature Combination												
SVR	**0.46**	**0.64**	**0.75**	**0.46**	**0.58**	**0.66**	**0.72**	**0.71**	0.62	0.60	**0.65**	**0.66**
IAA	0.54	0.58	0.61	0.53	0.56	0.60	0.68	0.68	0.64	0.67	0.63	0.64

Table 1: Results for various approaches to topic similarity. All correlations are significant $p < 0.001$. Underlined scores denote best performance of a single feature. Bold denotes best overall performance.

ate reliable estimates of tf.idf or co-document frequency.

ESA, one of the knowledge-based methods (Section 2.5), performs well and is comparable to (or in some cases better than) PMI. UKB does not perform particularly well because the topics often contain named entities that do not exist in WordNet. ESA is based on Wikipedia and does not suffer from this problem. Overall, metrics for computing topic similarity based on rich semantic resources (e.g. Wikipedia) are more appropriate than metrics based on the topic model itself because of the limited size of the training corpus.

Combining the features using SVR gives the best overall result for LDA (in the range 0.46-0.75) and CTM (0.60-0.72). However, the feature combination performs slightly lower than the best single feature in two cases when CTM is used (T=200, NYT and T=50, ukWAC). Analysis of the coefficients produced by the SVR in each fold demonstrated that including JSD and the Word Overlap reduce SVR performance. We repeated the experiments by removing these features[6] which resulted in higher correlations (0.64 and 0.65 respectively).

Another interesting observation is that using LDA the correlations of the various similarity metrics with human judgements increase with the number of topics for both corpora. This result is consistent with the findings of Stevens et al. (2012) that topic model coherence increases with the number of topics. Fewer topics makes the task of identifying similar topics more difficult because it is likely that they will contain some terms that do not relate to the topic's main subject. Correlations in CTM are more stable for different number of topics because of the nature of the model, the pairs have been generated using the topic graph which by definition contains correlated topics.

5 Conclusions

We explored the task of determining the similarity between pairs of automatically generated topics and described a range of approaches to the problem. We constructed a data set of pairs of topics generated by two topic models, LDA and CTM, together with human judgements of similarity. The data set was used to evaluate a wide range of approaches. The most interesting finding is the poor performance of the metrics based on word probability distributions previously used for this task. Our results demonstrate that word association measures, such as PMI, and state-of-the-art textual similarity metrics, such as ESA, are more appropriate.

[6]These features are useful for the other experiments since performance drops when they are removed.

References

Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca, and Aitor Soroa. 2009. A study on similarity and relatedness using distributional and wordnet-based approaches. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT '09)*, pages 19–27, Boulder, Colorado.

Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 13–22, Potsdam, Germany.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Language resources and evaluation*, 43(3):209–226.

David Blei and John Lafferty. 2006. Correlated topic models. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 147–154. MIT Press, Cambridge, MA.

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

David Blei, Lawrence Carin, and David Dunson. 2010. Probabilistic topic models. *Signal Processing Magazine, IEEE*, 27(6):55–65.

Allison June-Barlow Chaney and David M. Blei. 2012. Visualizing topic models. In *Proceedings of the Sixth International AAAI Conference on Weblogs and Social Media*, Dublin, Ireland.

Kenneth Ward Church and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, pages 76–83, Vancouver, British Columbia, Canada.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI '07)*, pages 1606–1611.

Brynjar Gretarsson, John O'Donovan, Svetlin Bostandjiev, Tobias Höllerer, Arthur Asuncion, David Newman, and Padhraic Smyth. 2012. TopicNets: Visual analysis of large text corpora with topic modeling. *ACM Trans. Intell. Syst. Technol.*, 3(2):23:1–23:26.

David Hall, Daniel Jurafsky, and Christopher D. Manning. 2008. Studying the history of ideas using topic models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 363–371, Honolulu, Hawaii.

Qi He, Bi Chen, Jian Pei, Baojun Qiu, Prasenjit Mitra, and Lee Giles. 2009. Detecting topic evolution in scientific literature: how can citations help? In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM '09)*, pages 957–966, Hong Kong, China.

Alexander Hinneburg, Rico Preiss, and René Schröder. 2012. TopicExplorer: Exploring document collections with topic models. In Peter A. Flach, Tijl Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 7524 of *Lecture Notes in Computer Science*, pages 838–841. Springer Berlin Heidelberg.

Dongwoo Kim and Alice Oh. 2011. Topic chains for understanding a news corpus. In *Computational Linguistics and Intelligent Text Processing*, pages 163–176. Springer.

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pages 577–584.

David Mimno, Hanna Wallach, Edmund Talley, Miriam Leenders, and Andrew McCallum. 2011. Optimizing semantic coherence in topic models. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 262–272, Edinburgh, Scotland, UK.

David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. 2009. Distributed algorithms for topic models. *J. Mach. Learn. Res.*, 10:1801–1828.

David Newman, Jey Han Lau, Karl Grieser, and Timothy Baldwin. 2010. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT '10)*, pages 100–108, Los Angeles, California.

Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing (EMNLP '09)*, pages 248–256, Singapore.

Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. 2012. Exploring topic coherence over many models and many topics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP '12)*, pages 952–961, Jeju Island, Korea.

Vladimir N Vapnik. 1998. *Statistical learning theory*. Wiley, New York.

Xiang Wang, Kai Zhang, Xiaoming Jin, and Dou Shen. 2009. Mining common topics from multiple asynchronous text streams. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining (WSDM '09)*, pages 192–201, Barcelona, Spain.

Projecting the Knowledge Graph to Syntactic Parsing

Andrea Gesmundo and **Keith B. Hall**
Google, Inc.
{agesmundo,kbhall}@google.com

Abstract

We present a syntactic parser training paradigm that learns from large scale Knowledge Bases. By utilizing the Knowledge Base context only during training, the resulting parser has no inference-time dependency on the Knowledge Base, thus not decreasing the speed during prediction. Knowledge Base information is injected into the model using an extension to the Augmented-loss training framework. We present empirical results that show this approach achieves a significant gain in accuracy for syntactic categories such as coordination and apposition.

1 Introduction

Natural Language Processing systems require large amounts of world knowledge to achieve state-of-the-art performance. Leveraging Knowledge Bases (KB) provides allows us to inject human curated world-knowledge into our systems. As these KBs have increased in size, we are now able to leverage this information to improve upon the state-of-the-art. Large scale KB have been developed rapidly in recent years, adding large numbers of entities and relations between the entities. Such entities can be of any kind: an object, a person, a place, a company, a book, etc. Entities and relations are stored in association with relevant data that describes the particular entity or relation; for example, the name of a book, it's author, other books by the same author, etc.. Large scale KB annotation efforts have focused on the collection of both current and historical entities, but are biased towards the contemporary entities.

Of the many publicly available KBs, we focus this study on the use of Freebase[1]: a large collaborative Knowledge Base composed and updated by a member community. Currently it contains roughly 40 million entities and 1.1 billion relations.

The aim of the presented work is to use the information provided by the KB to improve the accuracy of the statistical dependency parsing task (Kubler et al., 2009). In particular we focus on the recognition of relations such as coordination and apposition. This choice is motivated by the fact that the KB stores information about real-world entities while many of the errors associated with coordination and apposition is the lack of knowledge of these real-world entities.

We begin by defining the task (section 2). Following, we present the modified augmented-loss training framework (section 3). In section 4, we define how the Knowledge Base data is integrated into the training process. Finally, we discuss the empirical results (section 5).

2 Task

Apposition is a relation between two adjacent noun-phrases, where one noun-phrase specifies or modifying the other. For example, in the sentence "My friend Anna", the nouns "friend" and "Anna" are in apposition. Coordination between nouns relates two or more elements of the same kind. The coordination is often signaled by the appearance of a coordinating conjunction. For example, in the sentence "My friend and Anna", the nouns "friend" and "Anna" are in coordination. The semantic difference between the two relations is that the nouns in apposition refer to the same entity,

[1] www.freebase.com

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 28–32,
Gothenburg, Sweden, April 26-30 2014.

while the nouns in coordination refer to distinct entities of the same kind or sharing some properties.

Statistical parsers are inaccurate in classifying relations involving proper nouns that appear rarely in the training set. In the sentence:

"They invested in three companies, Google, Microsoft, and Yahoo."

"companies" is in apposition with the coordination "Google, Microsoft, and Yahoo". By integrating the information provided by a large scale KB into the syntactic parser, we attempt to increase the ability to disambiguate the relations involving these proper nouns, even if the parser has been trained on a different domain.

3 Model

We present a Syntactic Parsing model that learns from the KB. An important constraint that we impose, is that the speed of the Syntactic Parser must not decrease when this information is integrated. As the queries to the KB would significantly slow down the parser, we limit querying the KB to training. This constraint reduces the impact that the KB can have on the accuracy, but allows us to design a parser that can be substituted in any setting, even in the absence of the KB.

We propose a solution based on the Augmented-loss framework (Hall et al., 2011a). Augmented-loss is a training framework for structured prediction tasks such as parsing. It can be used to extend a standard objective function with additional loss-functions and be integrated with the structured perceptron training algorithm. The input is enriched with multiple datasets each associated with a loss function. The algorithm iterates over the datasets triggering parameter updates whenever the loss function is positive.

Loss functions return a positive value if the predicted output is "worse" than the gold standard. Augmented-loss allows for the inclusion of multiple objective functions, either based on intrinsic parsing quality or task-specific extrinsic measures of quality. In the original formalization, both the intrinsic and extrinsic losses require gold standard information. Thus, each dataset must specify a gold standard output for each input.

We extend the Augmented-loss framework to apply it when the additional dataset gold-standard is unknown. Without the gold standard, it is not possible to trigger updates using a loss function. Instead, we use a *sampling function*, $S(\cdot)$, that is defined such that: if $\hat{y}$ is a candidate parse tree, then $S(\hat{y})$ returns a parse tree that is guaranteed to be "not worse" than $\hat{y}$. In other words:

$$L^S(\hat{y}, S(\hat{y})) \geq 0 \quad (1)$$

Where the $L^S(\cdot)$ is the *implicit loss function*. This formalization will allow us to avoid stating explicitly the loss function. Notice that $S(\hat{y})$ is not guaranteed to be the "best" parse tree. It can be any parse tree in the search space that is "not worse" than $\hat{y}$. $S(\hat{y})$ can represent an incremental improvement over $\hat{y}$.

Algorithm 1 Augmented-loss extension

1: {Input loss function: $L(\cdot)$}
2: {Input sample function: $S(\cdot)$}
3: {Input data sets}:
4: $D^L = \{d_i^L = (x_i^L, y_i^L) \mid 1 \leq i \leq N^L\}$
5: $D^S = \{d_i^S = (x_i^S) \mid 1 \leq i \leq N^S\}$
6: $\theta = \vec{0}$
7: **repeat**
8: **for** $i = 1 \ldots N^L$ **do**
9: $\hat{y} = F_\theta(x_i^L)$
10: **if** $L(\hat{y}, y_i^L) > 0$ **then**
11: $\theta = \theta + \Phi(y_i^L) - \Phi(\hat{y})$
12: **end if**
13: **end for**
14: **for** $i = 1 \ldots N^S$ **do**
15: $\hat{y} = F_\theta(x_i^S)$
16: $y^* = S(\hat{y})$
17: $\theta = \theta + \Phi(y^*) - \Phi(\hat{y})$
18: **end for**
19: **until** converged
20: {Return model θ}

Algorithm 1 summarizes the extension to the Augmented-loss algorithm.

The algorithm takes as input: the loss function $L(\cdot)$; the sample function $S(\cdot)$; the loss function data samples D^L; and the sample function data samples D^S. Notice that D^L specifies the gold standard parse y_i^L for each input sentence x_i^L. While, D^S specifies only the input sentence x_i^S.

The model parameter are initialized to the zero vector (line 6). The main loop iterates until the model reaches convergence (lines 7-19). After which the model parameters are returned.

The first inner loop iterates over D^L (lines 8-13) executing the standard on-line training. The candidate parse, $\hat{y}$, for the current input sentence,

x_i^L, is predicted given the current model parameters, θ (line 9). In the structured perceptron setting (Collins and Roark, 2004; Daumé III et al., 2009), we have that:

$$F_\theta(x) = argmax_{y \in \mathcal{Y}} \ \theta \cdot \Phi(y) \qquad (2)$$

Where $\Phi(\cdot)$ is the mapping from a parse tree y to a high dimensional feature space. Then, the algorithm tests if the current prediction is wrong (line 10). In which case the model is updated promoting features that fire in the gold-standard $\Phi(y_i^L)$, and penalizing features that fire in the predicted output, $\Phi(\hat{y})$ (line 11).

The second inner loop iterates over D^S (lines 14-18). First, the candidate parse, $\hat{y}$, is predicted (line 15). Then the sample parse, y^*, is produced by the sample function (line 16). Finally, the parameters are updated promoting the features of y^*. The updates are triggered without testing if the loss is positive, since it is guaranteed that $L^S(\hat{y}, y^*) \geq 0$. Updating in cases where $L^S(\hat{y}, y^*) = 0$ does not harm the model. To optimize the algorithm, updates can be avoided when $\hat{y} = y^*$.

In order to simplify the algorithmic description, we define the algorithm with only one loss function and one sample function, and we formalized it for the specific task we are considering. This definitions can be trivially generalized to integrate multiple loss/sample functions and to be formalized for a generic structured prediction task. This generalization can be achieved following the guidelines of (Hall et al., 2011a). Furthermore, we defined the algorithm such that it first iterates over D^L and then over D^S. In practice, the algorithm can switch between the data sets with a desired frequency by using a scheduling policy as described in (Hall et al., 2011a). For the experiments, we trained on 8 samples of D^L followed by 1 samples of D^S, looping over the training sets.

4 Sample Function

We integrate the Knowledge Base data into the training algorithm using a sampling function. The idea is to correct errors in the candidate parse by using the KB. The sample function corrects only relations among entities described in the KB. Thus, it returns a better or equal parse tree that may still contain errors. This is sufficient to guarantee the constraint on the implicit loss function (equation 1).

The sample function receives as input the candidate dependency parse and the input sentence enriched with KB annotation. Then, it corrects the labels of each arc in the dependency tree connecting two entities. The labels are corrected according to the predictions produced by a classifier. As classifier we use a standard multi-class perceptron (Crammer and Singer, 2003). The classifier is trained in a preprocessing step on a parsed corpus enriched with KB data. The features used by the classifier are:

- Lexical features of the head and modifier.
- Sentence level features: words distance between head and modifier; arc direction (L/R); neighboring words.
- Syntactic features: POS and syntactic label of head and modifier and modifier's left sibling.
- Knowledge Base features: types defined for entities and for their direct relations.

5 Experiments

The primary training corpus is composed of manually annotated sentences with syntactic tress which are converted to dependency format using the Stanford converter v1.6 (de Marneffe et al., 2006). We run experiments using 10k sentences or 70k sentences from this corpus. The test set contains 16k manually syntactically annotated sentences crawled from the web. The test and train sets are from different domains. This setting may degrade the parser accuracy in labelling out-of-domain entities, as we discussed in section 2. Thus, we use web text as secondary training set to be used for the Augmented-loss loss sample training. Web text is available in any quantity, and we do not need to provide gold-standard parses in order to integrate it in the Augmented-loss sample training. The classifier is trained on 10k sentences extracted from news text which has been automatically parsed. We chose to train the classifier on news data as the quality of the automatic parses is much higher than on general web text. We do this despite the fact that we will apply the classifier to a different domain (the web text).

As dependency parser, we use an implementation of the transition-based dependency parsing framework (Nivre, 2008) with the arc-eager transition strategy. The part of Augmented-loss training based on the standard loss function, applies

Training set size	Model	appos F1	conj F1	LAS	UAS
70k sentences	Baseline	54.36	83.72	79.55	83.50
	Augmented-loss	55.64	84.47	79.71	83.71
10k sentences	Baseline	45.13	80.36	75.99	86.02
	Augmented-loss	48.06	81.63	76.16	86.18

Table 1: Accuracy Comparison.

the perceptron algorithm as in (Zhang and Clark, 2008) with a beam size of 16. The baseline is the same model but trained only the primary training corpus without Augmented-loss.

Table 1 reports the results of the accuracy comparison. It reports the metrics for Labeled Attachment Score (LAS) and Unlabeled Attachment Score (UAS) to measure the overall accuracy. The syntactic classes that are affected the most are apposition (appos) and conjunction (conj). On the development set we measured that the percentage of arcs connecting 2 entities that are labeled as conjunction is 36.11%. While those that are labelled as apposition is 25.06%. Each of the other 40 labels cover a small portion of the remaining 38.83%.

Training the models with the full primary training corpus (70k sentences), shows a significant gain for the Augmented-loss model. Apposition F1 gains 1.28, while conjunction gains 0.75. The LAS gain is mainly due to the gain of the two mentioned classes. It is surprising to measure a similar gain also for the unlabeled accuracy. Since the classifier can correct the label of an arc but never change the structure of the parse. This implies that just by penalizing a labeling action, the model learns to construct better parse structures.

Training the model with 10k sentences shows a significantly bigger gain on all the measures. This results shows that, in cases where the set of labeled data is small, this approach can be applied to integrate in unlimited amount of unlabeled data to boost the learning.

6 Related Work

As we mentioned, Augmented-loss (Hall et al., 2011a; Hall et al., 2011b) is perhaps the closest to our framework. Another difference with its original formalization is that it was primarily aimed to cases where the additional weak signal is precisely what we wish to optimize. Such as cases where we wish to optimize parsing to be used as an input to a downstream natural language processing tasks and the accuracies to be optimized are those of the downstream task and not directly the parsing accuracy. While our work is focused on integrating additional data in a semi-supervised fashion with the aim of improving the primary task's accuracy and/or adapt it to a different domain.

Another similar idea is (Chang et al., 2007) which presents a constraint driven learning. In this study, they integrate a weak signal into the training framework with the aim to improve the structured prediction models on the intrinsic evaluation metrics.

7 Conclusion

We extended the Augmented-loss framework defining a method for integrating new types of signals that require neither gold standard data nor an explicit loss function. At the same time, they allow the integration of additional information that can inform training to learn for specific types of phenomena.

This framework allows us to effectively integrate large scale KB in the training of structured prediction tasks. This approach integrates the data at training time without affecting the prediction time.

Experiments on syntactic parsing show that a significant gain for categories that model relation between entities defined in the KB.

References

Ming-Wei Chang, Lev Ratinov, and Dan Roth. 2007. Guiding semi-supervision with constraint-driven learning. In *ACL '07: Proceedings of the 45th Conference of the Association for Computational Linguistics*.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *ACL '04: Proceedings of the 42rd Conference of the Association for Computational Linguistics*.

Koby Crammer and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research*, 3:951–991.

Hal Daumé III, John Langford, and Daniel Marcu. 2009. Search-based structured prediction. *Submitted to Machine Learning Journal*.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure trees. In *LREC*.

Keith Hall, Ryan McDonald, Jason Katz-brown, and Michael Ringgaard. 2011a. Training dependency parsers by jointly optimizing multiple objectives. In *EMNLP '11: Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Keith Hall, Ryan McDonald, and Slav Petrov. 2011b. Training structured prediction models with extrinsic loss functions. In *Domain Adaptation Workshop at NIPS*, October.

Sandra Kubler, Ryan McDonald, and Joakim Nivre. 2009. Dependency parsing. In *Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers*.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. volume 34, pages 513–553.

Yue Zhang and Stephen Clark. 2008. A tale of two parsers: Investigating and combining graph-based and transition-based dependency parsing. In *EMNLP '08: Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 562–571.

A Vague Sense Classifier for Detecting Vague Definitions in Ontologies

Panos Alexopoulos
iSOCO S.A.
Madrid, Spain
palexopoulos@isoco.com

John Pavlopoulos
Department of Informatics,
Athens University of Economics and Business
Athens, Greece
annis@aueb.gr

Abstract

Vagueness is a common human knowledge and linguistic phenomenon, typically manifested by predicates that lack clear applicability conditions and boundaries such as *High*, *Expert* or *Bad*. In the context of ontologies and semantic data, the usage of such predicates within ontology element definitions (classes, relations etc.) can hamper the latter's quality, primarily in terms of shareability and meaning explicitness. With that in mind, we present in this paper a vague word sense classifier that may help both ontology creators and consumers to automatically detect vague ontology definitions and, thus, assess their quality better.

1 Introduction

Vagueness is a common human knowledge and language phenomenon, typically manifested by terms and concepts like *High*, *Expert*, *Bad*, *Near* etc., and related to our inability to precisely determine the extensions of such concepts in certain domains and contexts. That is because vague concepts have typically blurred boundaries which do not allow for a sharp distinction between the entities that fall within their extension and those that do not (Hyde, 2008) (Shapiro, 2006). For example, some people are borderline tall: not clearly *"tall"* and not clearly *"not tall"*.

Ontologies, in turn, are formal shareable conceptualizations of domains, describing the meaning of domain aspects in a common, machine-processable form by means of concepts and their interrelations (Chandrasekaran et al., January February 1999). As such, they are widely used for the production and sharing of structured data and knowledge that can be commonly understood among human and software agents.

When building ontologies and semantic data, engineers and domain experts often use predicates that are vague. While this is not always an intentional act, the use of such predicates influences in a negative way the comprehension of this data by other parties and limits their value as a reusable source of knowledge (Alexopoulos et al., 2013). The reason is the subjective interpretation of vague definitions that can cause **disagreements** among the people who develop, maintain or use a vague ontology. In fact, as shown in (Alexopoulos et al., 2013), vagueness in ontologies can be a source of problems in scenarios involving i) structuring data with a vague ontology (where disagreements among experts on the validity of vague statements may occur), ii) utilizing vague facts in ontology-based systems (where reasoning results might not meet users' expectations) and iii) integrating vague semantic information (where the merging of particular vague elements can lead to data that will not be valid for all its users).

In this context, our goal in this paper is to enable ontology producers (engineers and domain experts) as well as consumers (i.e., practitioners who want to reuse ontologies and semantic data) to detect, in an automatic way, ontology element definitions that are potentially vague. Such a detection will help ontology creators build more comprehensible and shareable ontologies (by refining, eliminating or just documenting vague definitions) and consumers assess, in an easier way, their usability and quality before deciding to use it.

Our approach towards such a detection involves training a classifier that may distinguish between vague and non-vague term word senses and using it to determine whether a given ontology element definition is vague or not. For example, the definition of the ontology class *"StrategicClient"* as *"A client that has a high value for the company"* is (and should be) characterized as vague while the definition of *"AmericanCompany"* as *"A com-*

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 33–37,
Gothenburg, Sweden, April 26-30 2014.

pany that has legal status in the Unites States" is not. The classifier is trained in a supervised way, using vague and non-vague sense examples, carefully constructed from $WordNet$.

The structure of the rest of the paper is as follows. In the next section we briefly present related work while in section 3 we describe in detail our vague sense classifier, including the training data we used and the evaluation we performed. Section 4 describes the results of applying the classifier in an a publicly available ontology, illustrating its usefulness as an ontology evaluation tool. Finally, section 5 summarizes our work and outlines its future directions.

2 Related Work

The phenomenon of vagueness in human language and knowledge has been studied from a logic and philosophical point of view in a number of works (Hyde, 2008) (Shapiro, 2006) and different theories and paradigms have been proposed to accommodate it, including supervaluationism (Keefe, 2008), many-valued logic and fuzzy logic (Klir and Yuan, 1995). Moreover, in the context of ontologies, one may find several works focusing on acquisition, conceptualization and representation of vague knowledge, mainly following a fuzzy logic based approach (Bobillo and Straccia, 2011) (Stoilos et al., 2008) (Abulaish, 2009). Nevertheless all these approaches rely on manual identification and analysis of vague terms and concepts by domain experts and, to the best of our knowledge, no work attempts to automate this task.

Another set of related work consists of approaches for subjectivity and polarity labeling of word senses (Wiebe and Riloff, 2005) (Wiebe and Mihalcea, 2006) (Wilson et al., 2005) (Su and Markert, 2008) (Esuli and Sebastiani, 2006) (Akkaya et al., 2011). While vagueness is related to both phenomena (as polarized words are often vague and vague words are typically subjective), it is not exactly the same as these (e.g., subjective statements do not always involve vagueness) and, thus, requires specialized treatment. To illustrate that, we compare in subsequent sections our vague sense classifier with the subjective sense classifier of (Wilson et al., 2005), showing that the former performs significantly better than the latter.

3 Supervised Classification for Vague Term Detection

3.1 Data

We created a dataset of 2,000 adjective senses, collected from $WordNet$, such that 1,000 of them had a vague definition and the the rest a non vague definition. A sample of these senses is shown in Table 1 while the whole dataset, which to the best of our knowledge is the first of its kind, is publicly available for further research[1].

The dataset was constructed by an ontology expert. As the task of classifying a text as vague or not can be quite subjective, we asked from two other human judges to annotate a subset of the dataset's definitions (100), and we measured inter-annotator agreement between all three. We found mean pairwise JPA (Joint Probability of Agreement) equal to 0.81 and mean pairwise K (Cohen's Kappa) equal to 0.64, both of which indicate a reasonable agreement.

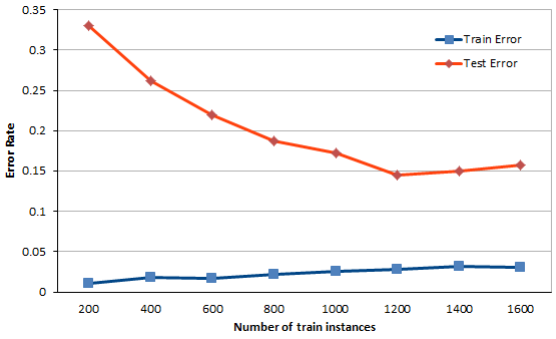


Figure 1: Train and test error rate, per number of training instances.

3.2 Training and Evaluation

We used the first 80% of the data (i.e., 800 vague and 800 non vague instances) to train a multinomial Naive Bayes classifier.[2] We removed stop words and we used the bag of words assumption to represent each instance.[3] The remaining 20% of the data (i.e., 200 vague and 200 non vague instances) was used as a test set. Accuracy was found to be 84%, which is considerably high. In Figure 1, is shown the error rate on the test and train data, as we increase the number of training instances. We see that the two curves, initially,

[1] http://glocal.isoco.net/datasets/VagueSynsets.zip

[2] We used the implementation of Scikit-Learn found at `http://scikit-learn.org/stable/`.

[3] We used the list of stopwords provided by Scikit-Learn.

Vague Adjectives	Non Vague Adjectives
Abnormal: not normal, not typical or usual or regular or conforming to a norm	Compound: composed of more than one part
Impenitent: impervious to moral persuasion	Biweekly: occurring every two weeks
Notorious: known widely and usually unfavorably	Irregular: falling below the manufacturer's standard
Aroused: emotionally aroused	Outermost: situated at the farthest possible point from a center
Yellowish: of the color intermediate between green and orange in the color spectrum, of something resembling the color of an egg yolk	Unfeathered: having no feathers

Table 1: Sample Vague and Non-Vague Adjective Senses

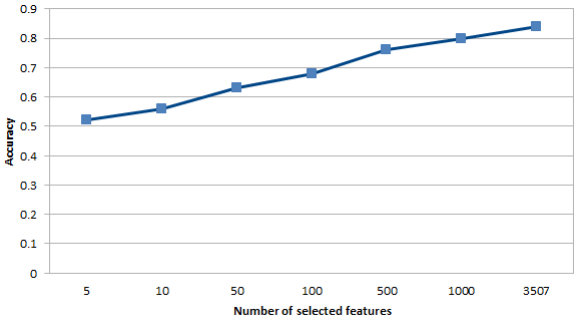


Figure 2: Accuracy on the test data, per number of selected features.

have a big gap between them, but this is progressively reduced. However, more (or more complicated) features could be beneficial; we intend to study this further in the future.

We also examined the hypothesis of the existence of a small set of words that are often found in vague definitions, but not in definitions which are not vague, as then it would be very easy for a system to use these words and discriminate between the two classes. To do this, we performed feature selection with the chi-squared statistic for various number of features and computed the accuracy (i.e., one minus the error rate). As we show in Figure 2, accuracy for only 5 selected features is 50%, which is the same as if we selected class in random. However, by increasing the number of selected features, accuracy increases significantly. This shows that there is not a subset of words which could be used to discriminate between the two classes; by contrast, most of the words play their role. Again, this is something to be further studied in future research.

Finally, in order to verify our intuition that vagueness is not the same phenomenon as subjectiveness (as we suggested in section 2), we used the subjective sense classifier of (Wilson et al., 2005) to classify the data of section 3.1 as subjective or objective, assuming that vague senses are subjective while non-vague ones objective. The particular classifier is part of the OpinionFinder[4] system and the results of its application in the 2000 adjective senses of our dataset were as follows. From the 1000 vague senses, only 167 were classified as subjective while from the 1000 non-vague ones 993. These numbers do no reflect of course the quality of OpinionFinder as a subjectivity detection system, they merely illustrate the fact that treating vagueness in the same way as subjectiveness is not really effective and, thus, more dedicated, vagueness-specific work is needed.

4 Use Case: Detecting Vagueness in CiTO Ontology

To evaluate the effectiveness and potential of our classifier for detecting vague ontological definitions, we considered a publicly available ontology called CiTO[5]. CiTO is an ontology that enables characterization of the nature or type of citations and consists primarily of relations, many of which are vague (e.g. the relation *cito:plagiarizes*). In order to compare the experts' vague/non-vague classification with the output of our system, we worked as follows. We selected 44 relations from CiTO (making sure to avoid duplications by e.g. avoiding having both a relation and its inverse) and we had again 3 human judges manually classify them as vague or not. In the end we got 27 vague

[4]http://mpqa.cs.pitt.edu/opinionfinder/
[5]http://purl.org/spar/cito/

Vague Relations	Non Vague Relations
plagiarizes: A property indicating that the author of the citing entity plagiarizes the cited entity, by including textual or other elements from the cited entity without formal acknowledgement of their source.	*sharesAuthorInstitutionWith*: Each entity has at least one author that shares a common institutional affiliation with an author of the other entity.
citesAsAuthority: The citing entity cites the cited entity as one that provides an authoritative description or definition of the subject under discussion.	*providesDataFor*: The cited entity presents data that are used in work described in the citing entity.
speculatesOn: The citing entity speculates on something within or related to the cited entity, without firm evidence.	*retracts*: The citing entity constitutes a formal retraction of the cited entity.
supports: The citing entity provides intellectual or factual support for statements, ideas or conclusions presented in the cited entity.	*includesExcerptFrom*: The citing entity includes one or more excerpts from the cited entity.
refutes: The citing entity refutes statements, ideas or conclusions presented in the cited entity.	*citesAsSourceDocument:* The citing entity cites the cited entity as being the entity from which the citing entity is derived, or about which the citing entity contains metadata.

Table 2: Sample Vague and Non-Vague Relations in CiTO

relations and 17 non-vague, a sample of which is shown in Table 2.

Then we applied the trained vagueness classifier of the previous section on the textual definitions of the relations. The results of this were highly encouraging; 36/44 (82%) relations were correctly classified as vague/non-vague with 74% accuracy for vague relations and 94% for non-vague ones. Again, for completeness, we classified the same relations with OpinionFinder (as in the previous section), in order to check if subjectivity classification is applicable for vagueness. The results of this were consistent to the ones reported in the previous section with the Wordnet data: 18/44 (40%) overall correctly classified relations with 94% accuracy for non-vague relations but only 7% for vague ones.

5 Conclusions and Future Work

In this paper we considered the problem of automatically detecting vague definitions in ontologies and we developed a vague word sense classifier using training data from $Wordnet$. Experiments with both $Wordnet$ word senses and real ontology definitions, showed a considerably high accuracy of our system, thus verifying our intuition that vague and non-vague senses can be separable. We do understand that vagueness is a quite complex phenomenon and the approach we have followed in this paper rather simple. Yet, exactly because of its simplicity, we believe that it can be a very good baseline for further research in this particular area. The vague/non-vague sense dataset we provide will be also very useful for that purpose.

Our future work comprises two main directions. On the one hand, as we mentioned in the introduction, we intend to incorporate the current classifier into an ontology analysis tool that will help ontology engineers and users detect vague definitions in ontologies and thus assess their quality better. On the other hand, we want to further study the phenomenon of vagueness as manifested in textual information, improve our classifer and see whether it is possible to build a vague sense lexicon, similar to lexicons that have already been built for subjectivity and sentiment analysis.

Acknowledgments

The research leading to this results has received funding from the People Programme (Marie Curie Actions) of the European Union's 7th Framework Programme P7/2007-2013 under REA grant agreement n^o 286348.

References

M. Abulaish. 2009. An ontology enhancement framework to accommodate imprecise concepts and relations. *Journal of Emerging Technologies in Web Intelligence*, 1(1).

C. Akkaya, J. Wiebe, A. Conrad, and R. Mihalcea. 2011. Improving the impact of subjectivity word sense disambiguation on contextual opinion analysis. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, CoNLL '11, pages 87–96, Stroudsburg, PA, USA. Association for Computational Linguistics.

P. Alexopoulos, B. Villazon-Terrazas, and Pan J.Z. Pan. 2013. Towards vagueness-aware semantic data. In Fernando Bobillo, Rommel N. Carvalho, Paulo Cesar G. da Costa, Claudia d'Amato, Nicola Fanizzi, Kathryn B. Laskey, Kenneth J. Laskey, Thomas Lukasiewicz, Trevor Martin, Matthias Nickles, and Michael Pool, editors, *URSW*, volume 1073 of *CEUR Workshop Proceedings*, pages 40–45. CEUR-WS.org.

F. Bobillo and U. Straccia. 2011. Fuzzy ontology representation using owl 2. *International Journal of Approximate Reasoning*, 52(7):1073–1094, October.

B. Chandrasekaran, J. Josephson, and R. Benjamins. January - February 1999. What are ontologies and why do we need them? *IEEE Intelligent Systems*, 14(1):Page 20–26.

A. Esuli and F. Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06*, pages 417–422.

D. Hyde. 2008. *Vagueness, Logic and Ontology*. Ashgate New Critical Thinking in Philosophy.

R. Keefe. 2008. Vagueness: Supervaluationism. *Philosophy Compass*, 3:315–324.

G. Klir and B. Yuan. 1995. *Fuzzy Sets and Fuzzy Logic, Theory and Applications*. Prentice Hall.

S. Shapiro. 2006. *Vagueness in Context*. Oxford University Press.

G. Stoilos, G. Stamou, J.Z. Pan, N. Simou, and V. Tzouvaras. 2008. Reasoning with the Fuzzy Description Logic f-SHIN: Theory, Practice and Applications. pages 262–281.

F. Su and K. Markert. 2008. From words to senses: A case study of subjectivity recognition. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 825–832, Stroudsburg, PA, USA. Association for Computational Linguistics.

J. Wiebe and R. Mihalcea. 2006. Word sense and subjectivity. In *Proceedings of COLING-ACL 2006*.

J. Wiebe and E. Riloff. 2005. Creating subjective and objective sentence classifiers from unannotated texts. In *In CICLing2005*, pages 486–497.

T. Wilson, P. Hoffmann, S. Somasundaran, J. Kessler, J. Wiebe, Y. Choi, C. Cardie, E. Riloff, and S. Patwardhan. 2005. Opinionfinder: A system for subjectivity analysis. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, HLT-Demo '05, pages 34–35, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chasing Hypernyms in Vector Spaces with Entropy

Enrico Santus
Dept. of Chinese and Bilingual Studies
The Hong Kong Polytechnic University, Hong Kong
e.santus@connect.polyu.hk

Alessandro Lenci
CoLing Lab – Dept. of Philology, Literature, and Linguistics
University of Pisa, Italy
alessandro.lenci@ling.unipi.it

Qin Lu
Dept. of Computing
The Hong Kong Polytechnic University, Hong Kong
csluqin@comp.polyu.edu.hk

Sabine Schulte im Walde
Inst. for Natural Language Processing
University of Stuttgart
Germany
schulte@ims.uni-stuttgart.de

Abstract

In this paper, we introduce *SLQS*, a new entropy-based measure for the unsupervised identification of hypernymy and its directionality in Distributional Semantic Models (DSMs). *SLQS* is assessed through two tasks: (i.) identifying the hypernym in hyponym-hypernym pairs, and (ii.) discriminating hypernymy among various semantic relations. In both tasks, *SLQS* outperforms other state-of-the-art measures.

1 Introduction

In recent years, Distributional Semantic Models (DSMs) have gained much attention in computational linguistics as unsupervised methods to build lexical semantic representations from corpus-derived co-occurrences encoded as distributional vectors (Sahlgren, 2006; Turney and Pantel, 2010). DSMs rely on the *Distributional Hypothesis* (Harris, 1954) and model lexical semantic similarity as a function of distributional similarity, which is most commonly measured with the *vector cosine* (Turney and Pantel, 2010). DSMs have achieved impressive results in tasks such as synonym detection, semantic categorization, etc. (Padó and Lapata, 2007; Baroni and Lenci, 2010).

One major shortcoming of current DSMs is that they are not able to discriminate among different types of semantic relations linking distributionally similar lexemes. For instance, the nearest neighbors of *dog* in vector spaces typically include hypernyms like *animal*, co-hyponyms like *cat*, meronyms like *tail*, together with other words semantically related to *dog*. DSMs tell us how similar these words are to *dog*, but they do not give us a principled way to single out the items linked by a specific relation (e.g., hypernyms).

Another related issue is to what extent distributional similarity, as currently measured by DSMs, is appropriate to model the semantic properties of a relation like hypernymy, which is crucial for Natural Language Processing. Similarity is by definition a symmetric notion (*a* is similar to *b* if and only if *b* is similar to *a*) and it can therefore naturally model symmetric semantic relations, such as synonymy and co-hyponymy (Murphy, 2003). It is not clear, however, how this notion can also model hypernymy, which is asymmetric. In fact, it is not enough to say that *animal* is distributionally similar to *dog*. We must also account for the fact that *animal* is semantically broader than *dog*: every *dog* is an *animal*, but not every *animal* is a *dog*.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 38–42,
Gothenburg, Sweden, April 26-30 2014.

In this paper, we introduce *SLQS*, a new entropy-based distributional measure that aims to identify hypernyms by providing a distributional characterization of their *semantic generality*. We assess it with two tasks: (i.) the identification of the broader term in hyponym-hypernym pairs (*directionality task*); (ii.) the discrimination of hypernymy among other semantic relations (*detection task*). Given the centrality of hypernymy, the relevance of the themes we address hardly needs any further motivation. Improving the ability of DSMs to identify hypernyms is in fact extremely important in tasks such as Recognizing Textual Entailment (RTE) and ontology learning, as well as to enhance the cognitive plausibility of DSMs as general models of the semantic lexicon.

2 Related work

The problem of identifying asymmetric relations like hypernymy has so far been addressed in distributional semantics only in a limited way (Kotlerman et al., 2010) or treated through semi-supervised approaches, such as pattern-based approaches (Hearst, 1992). The few works that have attempted a completely unsupervised approach to the identification of hypernymy in corpora have mostly relied on some versions of the *Distributional Inclusion Hypothesis* (DIH; Weeds and Weir, 2003; Weeds et al., 2004), according to which the contexts of a narrow term are also shared by the broad term.

One of the first proposed measures formalizing the DIH is *WeedsPrec* (Weeds and Weir, 2003; Weeds et al., 2004), which quantifies the weights of the features *f* of a narrow term *u* that are included into the set of features of a broad term *v*:

$$WeedsPrec(u,v) = \frac{\sum_{f \in F_u \cap F_v} w_u(f)}{\sum_{f \in F_u} w_u(f)}$$

where F_x is the set of features of a term *x*, and $w_x(f)$ is the weight of the feature *f* of the term *x*. Variations of this measure have been introduced by Clarke (2009), Kotlerman et al. (2010) and Lenci and Benotto (2012).

In this paper, we adopt a different approach, which is not based on DIH, but on the hypothesis that hypernyms are semantically more general than hyponyms, and therefore tend to occur in less informative contexts than hypernyms.

3 *SLQS*: A new entropy-based measure

DIH is grounded on an "extensional" definition of the asymmetric character of hypernymy: since the class (i.e., extension) denoted by a hyponym is included in the class denoted by the hypernym, hyponyms are expected to occur in a subset of the contexts of their hypernyms. However, it is also possible to provide an "intensional" definition of the same asymmetry. In fact, the typical characteristics making up the "intension" (i.e., concept) expressed by a hypernym (e.g., *move* or *eat* for *animal*) are semantically more general than the characteristics forming the "intension" of its hyponyms (e.g., *bark* or *has fur* for *dog*). This corresponds to the idea that superordinate terms like *animal* are less informative than their hyponyms (Murphy, 2002). From a distributional point of view, we can therefore expect that the most typical linguistic contexts of a hypernym are less informative than the most typical linguistic contexts of its hyponyms. In fact, contexts such as *bark* and *has fur* are likely to co-occur with a smaller number of words than *move* and *eat*. Starting from this hypothesis and using entropy as an estimate of context informativeness (Shannon, 1948), we propose *SLQS*, which measures the semantic generality of a word by the entropy of its statistically most prominent contexts.

For every term w_i we identify the *N* most associated contexts *c* (where *N* is a parameter empirically set to 50)[1]. The association strength has been calculated with *Local Mutual Information* (LMI; Evert, 2005). For each selected context *c*, we define its entropy *H(c)* as:

[1] *N=50* is the result of an optimization of the model against the dataset after trying the following suboptimal values: 5, 10, 25, 75 and 100.

$$H(c) = -\sum_{i=1}^{n} p(f_i|c) \cdot log_2\big(p(f_i|c)\big)$$

where $p(f_i|c)$ is the probability of the feature f_i given the context c, obtained through the ratio between the frequency of $<c, f_i>$ and the total frequency of c. The resulting values $H(c)$ are then normalized in the range 0-1 by using the Min-Max-Scaling (Priddy and Keller, 2005): $H_n(c)$. Finally, for each term w_i we calculate the median entropy E_{wi} of its N contexts:

$$E_{w_i} = Me_{j=1}^{N}\left(H_n(c_j)\right)$$

E_{w_i} can be considered as a *semantic generality index* for the term w_i: the higher E_{w_i}, the more semantically general w_i is. *SLQS* is then defined as the reciprocal difference between the semantic generality E_{w_1} and E_{w_2} of two terms w_1 and w_2:

$$SLQS(w_1, w_2) = 1 - \frac{E_{w_1}}{E_{w_2}}$$

According to this formula, $SLQS<0$, if $E_{w_1}>E_{w_2}$; $SLQS\simeq 0$, if $E_{w_1}\simeq E_{w_2}$; and $SLQS>0$, if $E_{w_1}<E_{w_2}$. *SLQS* is an asymmetric measure because, by definition, $SLQS(w_1,w_2)\neq SLQS(w_2,w_1)$ (except when w_1 and w_2 have exactly the same generality). Therefore, if $SLQS(w_1,w_2)>0$, w_1 is semantically less general than w_2.

4 Experiments and evaluation

4.1 The DSM and the dataset

For the experiments, we used a standard window-based DSM recording co-occurrences with the nearest 2 content words to the left and right of each target word. Co-occurrences were extracted from a combination of the freely available ukWaC and WaCkypedia corpora (with 1.915 billion and 820 million words, respectively) and weighted with LMI.

To assess *SLQS* we relied on a subset of *BLESS* (Baroni and Lenci, 2011), a freely-available dataset that includes 200 distinct English concrete nouns as target concepts, equally divided between living and non-living entities (e.g. `BIRD`, `FRUIT`, etc.). For each target concept, *BLESS* contains several *relata*, connected to it through one relation, such as co-hyponymy (`COORD`), hypernymy (`HYPER`), meronymy (`MERO`) or no-relation (`RANDOM-N`).[2]

Since *BLESS* contains different numbers of pairs for every relation, we randomly extracted a subset of 1,277 pairs for each relation, where 1,277 is the maximum number of `HYPER`-related pairs for which vectors existed in our DSM.

4.2 Task 1: Directionality

In this experiment we aimed at identifying the hypernym in the 1,277 hypernymy-related pairs of our dataset. Since the `HYPER`-related pairs in *BLESS* are in the order hyponym-hypernym (e.g. *eagle-bird*, *eagle-animal*, etc.), the hypernym in a pair (w_1,w_2) is correctly identified by *SLQS*, if $SLQS(w_1,w_2) > 0$. Following Weeds et al. (2004), we used word frequency as a baseline model. This baseline is grounded on the hypothesis that hypernyms are more frequent than hyponyms in corpora. Table 1 gives the evaluation results:

	SLQS	WeedsPrec	BASELINE
POSITIVE	1111	805	844
NEGATIVE	166	472	433
TOTAL	**1277**	**1277**	**1277**
PRECISION	87.00%	63.04%	66.09%

Table 1. Accuracy for Task 1.

As it can be seen in Table 1, *SLQS* scores a precision of 87% in identifying the second term of the test pairs as the hypernym. This result is particularly significant when compared to the one obtained by applying WeedsPrec (+23.96%). As it was also noticed by Geffet and Dagan (2005) with reference to a previous similar experiment performed on a different corpus (Weeds et al., 2004), the WeedsPrec precision in this task is comparable to the naïve baseline. *SLQS* scores instead a +20.91%.

[2] In these experiments, we only consider the *BLESS* pairs containing a noun relatum.

4.3 Task 2: Detection

The second experiment aimed at discriminating HYPER test pairs from those linked by other types of relations in *BLESS* (i.e., MERO, COORD and RANDOM-N). To this purpose, we assumed that hypernymy is characterized by two main properties: (i.) the hypernym and the hyponym are distributionally similar (in the sense of the *Distributional Hypothesis*), and (ii.) the hyponym is semantically less general than the hypernym. We measured the first property with the *vector cosine* and the second one with *SLQS*.

After calculating *SLQS* for all the pairs in our datasets, we set to zero all the negative values, that is to say those in which – according to *SLQS* – the first term is semantically more general than the second one. Then, we combined *SLQS* and *vector cosine* by their product. The greater the resulting value, the greater the likelihood that we are considering a hypernymy-related pair, in which the first word is a hyponym and the second word is a hypernym.

To evaluate the performance of *SLQS*, we used *Average Precision* (AP; Kotlerman et al., 2010), a method derived from Information Retrieval that combines precision, relevance ranking and overall recall, returning a value that ranges from 0 to 1. AP=1 means that all the instances of a relation are in the top of the rank, whereas AP=0 means they are in the bottom. AP is calculated for the four relations we extracted from *BLESS*. *SLQS* was also compared with *WeedsPrec* and *vector cosine*, again using frequency as baseline. Table 2 shows the results:

	HYPER	COORD	MERO	RANDOM
Baseline	0.40	0.51	0.38	0.17
Cosine	0.48	0.46	0.31	0.21
WeedsPrec	0.50	0.35	0.39	0.21
*SLQS * Cosine*	**0.59**	**0.27**	**0.35**	**0.24**

Table 2. AP values for Task 2.

The AP values show the performances of the tested measures on the four relations. The optimal result would be obtained scoring 1 for HYPER and 0 for the other relations.

The product between *SLQS* and *vector cosine* gets the best performance in identifying HYPER (+0.09 in comparison to *WeedsPrec*) and in discriminating it from COORD (-0.08 than *WeedsPrec*). It also achieves better results in discriminating MERO (-0.04 than *WeedsPrec*). On the other hand, it seems to get a slightly lower precision in discriminating RANDOM-N (+0.03 in comparison to *WeedsPrec*). The likely reason is that unrelated pairs might also have a fairly high semantic generality difference, slightly affecting the measure's performance. Figure 1 gives a graphic depiction of the performances. *SLQS* corresponds to the black line in comparison to the *WeedsPrec* (black borders, grey fill), the *vector cosine* (grey borders) and the baseline (grey fill).

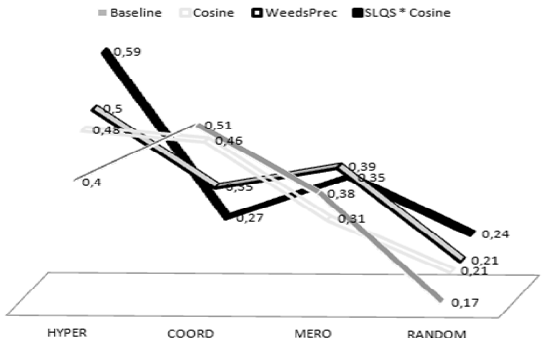


Figure 1. AP values for Task 2.

5 Conclusions and future work

In this paper, we have proposed *SLQS*, a new asymmetric distributional measure of semantic generality which is able to identify the broader term in a hypernym-hyponym pair and, when combined with *vector cosine*, to discriminate hypernymy from other types of semantic relations. The successful performance of *SLQS* in the reported experiments confirms that hyponyms and hypernyms are distributionally similar, but hyponyms tend to occur in more informative contexts than hypernyms. *SLQS* shows that an "intensional" characterization of hypernymy can be pursued in distributional terms. This opens up new possibilities for the study of semantic relations in DSMs. In further research, *SLQS* will also be tested on other datasets and languages.

References

Baroni, Marco and Lenci, Alessandro. 2010. "Distributional Memory: A general framework for corpus-based semantics". *Computational Linguistics*, Vol. 36 (4). 673-721.

Baroni, Marco and Lenci, Alessandro. 2011. "How we BLESSed distributional semantic evaluation". *Proceedings of the EMNLP 2011 Geometrical Models for Natural Language Semantics (GEMS 2011) Workshop*. Edinburg, UK. 1-10.

Clarke, Daoud. 2009. "Context-theoretic semantics for natural language: An overview". *Proceedings of the Workshop on Geometrical Models of Natural Language Semantics*. Athens, Greece. 112-119.

Evert, Stefan. 2005. *The Statistics of Word Cooccurrences*. Dissertation, Stuttgart University.

Geffet, Maayan and Dagan, Idan. 2005. "The Distributional Inclusion Hypotheses and Lexical Entailment". *Proceedings of 43rd Annual Meeting of the ACL*. Michigan, USA. 107-114.

Harris, Zellig. 1954. "Distributional structure". *Word*, Vol. 10 (23). 146-162.

Hearst, Marti A. 1992. "Automatic Acquisition of Hyponyms from Large Text Corpora". *Proceedings of the 14th International Conference on Computational Linguistics*. Nantes, France. 539-545.

Kotlerman, Lili, Dagan, Ido, Szpektor, Idan, and Zhitomirsky-Geffet, Maayan. 2010. "Directional Distributional Similarity for Lexical Inference". *Natural Language Engineering*, Vol. 16 (4). 359-389.

Lenci, Alessandro and Benotto, Giulia. 2012. "Identifying hypernyms in distributional semantic spaces". *SEM 2012 – The First Joint Conference on Lexical and Computational Semantics*. Montréal, Canada. Vol. 2. 75-79.

Murphy, Gregory L.. 2002. *The Big Book of Concepts*. The MIT Press, Cambridge, MA.

Murphy, M. Lynne. 2003. *Lexical meaning*. Cambridge University Press, Cambridge.

Padó, Sebastian and Lapata, Mirella. 2007. "Dependency-based Construction of Semantic Space Models". *Computational Linguistics*, Vol. 33 (2). 161-199.

Priddy, Kevin L. and Keller, Paul E. 2005. *Artificial Neural Networks: An Introduction*. SPIE Press - International Society for Optical Engineering, October 2005.

Sahlgren, Magnus. 2006. *The Word-Space Model: Using distributional analysis to represent syntagmatic and paradigmatic relations between words in high-dimensional vector spaces*. Ph.D. dissertation, Department of Linguistics, Stockholm University.

Shannon, Claude E. 1948. "A mathematical theory of communication". *Bell System Technical Journal*, Vol. 27. 379-423 and 623-656.

Turney, Peter D. and Pantel, Patrick. 2010. "From Frequency to Meaning: Vector Space Models of Semantics". *Journal of Articial Intelligence Research*, Vol. 37. 141-188.

Weeds, Julie and Weir, David. 2003. "A general framework for distributional similarity". *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*. Sapporo, Japan. 81-88.

Weeds, Julie, Weir, David and McCarthy, Diana. 2004. "Characterising measures of lexical distributional similarity". *Proceedings of COLING 2004*. Geneva, Switzerland.1015-1021.

Tight Integration of Speech Disfluency Removal into SMT

Eunah Cho **Jan Niehues** **Alex Waibel**
Interactive Systems Lab
Institute of Anthropomatics
Karlsruhe Institute of Technology, Germany
{eunah.cho,jan.niehues,alex.waibel}@kit.edu

Abstract

Speech disfluencies are one of the main challenges of spoken language processing. Conventional disfluency detection systems deploy a hard decision, which can have a negative influence on subsequent applications such as machine translation. In this paper we suggest a novel approach in which disfluency detection is integrated into the translation process.

We train a CRF model to obtain a disfluency probability for each word. The SMT decoder will then skip the potentially disfluent word based on its disfluency probability. Using the suggested scheme, the translation score of both the manual transcript and ASR output is improved by around 0.35 BLEU points compared to the CRF hard decision system.

1 Introduction

Disfluencies arise due to the spontaneous nature of speech. There has been a great deal of effort to detect disfluent words, remove them (Johnson and Charniak, 2004; Fitzgerald et al., 2009) and use the cleaned text for subsequent applications such as machine translation (MT) (Wang et al., 2010; Cho et al., 2013).

One potential drawback of conventional approaches is that the decision whether a token is a disfluency or not is a hard decision. For an MT system, this can pose a severe problem if the removed token was not in fact a disfluency and should have been kept for the correct translation. Therefore, we pass the decision whether a word is part of a disfluency or not on to the translation system, so that we can use the additional knowledge available in the translation system to make a more reliable decision. In order to limit the complexity, the search space is pruned prior to decoding and represented in a word lattice.

2 Related Work

Disfluencies in spontaneous speech have been studied from various points of view. In the noisy channel model (Honal and Schultz, 2003), it is assumed that clean text without any disfluencies has passed through a noisy channel. The clean string is retrieved based on language model (LM) scores and five additional models. Another noisy channel approach involves a phrase-level statistical MT system, where noisy tokens are translated into clean tokens (Maskey et al., 2006). A tree adjoining grammar is combined with this noisy channel model in (Johnson and Charniak, 2004), using a syntactic parser to build an LM.

Fitzgerald et al. (2009) present a method to detect speech disfluencies using a conditional random field (CRF) with lexical, LM, and parser information features. While previous work has been limited to the postprocessing step of the automatic speech recogition (ASR) system, further approaches (Wang et al., 2010; Cho et al., 2013) use extended CRF features or additional models to clean manual speech transcripts and use them as input for an MT system.

While ASR systems use lattices to encode hypotheses, lattices have been used for MT systems with various purposes. Herrmann et al. (2013) use lattices to encode different reordering variants. Lattices have also been used as a segmentation tactic for compound words (Dyer, 2009), where the segmentation is encoded as input in the lattice.

One of the differences between our work and previous work is that we integrate the disfluency removal into an MT system. Our work is not limited to the preprocessing step of MT, instead we use the translation model to detect and remove disfluencies. Contrary to other systems where detection is limited on manual transcripts only, our sys-

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 43–47,
Gothenburg, Sweden, April 26-30 2014.

tem shows translation performance improvements on the ASR output as well.

3 Tight Integration using Lattices

In this chapter, we explain how the disfluency removal is integrated into the MT process.

3.1 Model

The conventional translation of texts from spontaneous speech can be formulated as

$$\hat{e} = \arg\max_{e} p(e|\arg\max_{f_c} p(f_c|f)) \quad (1)$$

with

$$p(f_c|f) = \prod_{i=1}^{I} p(c_i|f_i) \quad (2)$$

where f_c denotes the clean string

$$f_c = \{f_1, \ldots, f_I \mid c_i = clean\} \quad (3)$$

for the disfluency decision class c of each token.

$$c \in \begin{cases} clean \\ disfluent \end{cases} \quad (4)$$

Thus, using the conventional models, disfluency removal is applied to the original, potentially noisy string in order to obtain the cleaned string first. This clean string is then translated.

The potential drawback of a conventional speech translation system is caused by the rough estimation in Equation 1, as disfluency removal does not depend on maximizing the translation quality itself. For example, we can consider the sentence *Use what you build, build what you use*. Due to its repetitive pattern in words and structure, the first clause is often detected as a disfluency using automatic means. To avoid this, we can change the scheme how the clean string is chosen as follows:

$$\hat{e} = \arg\max_{e,f_c}(p(e|f_c) \cdot p(f_c|f)) \quad (5)$$

This way a clean string which maximizes the translation quality is chosen. Thus, no instant decision is made whether a token is a disfluency or not. Instead, the disfluency probability of the token will be passed on to the MT process, using the log linear combination of the probabilities as shown in Equation 5.

In this work, we use a CRF (Lafferty et al., 2001) model to obtain the disfluency probability of each token.

Since there are two possible classes for each token, the number of possible clean sentences is exponential with regard to the sentence length. Thus, we restrict the search space by representing only the most probable clean source sentences in a word lattice.

3.2 CRF Model Training

In order to build the CRF model, we used the open source toolkit CRF++ (Kudoh, 2007). As unigram features, we use lexical and LM features adopted from Fitzgerald et al. (2009), and additional semantics-based features discussed in (Cho et al., 2013). In addition to the unigram features, we also use a bigram feature to model first-order dependencies between labels.

We train the CRF with four classes; `FL` for filler words, `RC` for (rough) copy, `NC` for non-copy and `0` for clean tokens. The class `FL` includes obvious filler words (e.g. *uh*, *uhm*) as well as other discourse markers (e.g. *you know*, *well* in English). The `RC` class covers identical or roughly similar repetitions as well as lexically different words with the same meaning. The `NC` class represents the case where the speaker changes what to speak about or reformulates the sentence and restarts the speech fragments. The disfluency probability P_d of each token is calculated as the sum of probabilities of each class.

3.3 Lattice Implementation

We construct a word lattice which encodes long-range reordering variants (Rottmann and Vogel, 2007; Niehues and Kolss, 2009). For translation we extend this so that potentially disfluent words can be skipped. A reordering lattice of the example sentence *Das sind die Vorteile, die sie uh die sie haben.* (En.gls: *These are the advantages, that you uh that you have.*) is shown in Figure 1, where words representing a disfluency are marked in bold letters. In this sentence, the part *die sie uh* was manually annotated as a disfluency, due to repetition and usage of a filler word.

Table 1 shows the P_d obtained from the CRF model for each token. As expected, the words *die sie uh* obtain a high P_d from the CRF model.

In order to provide an option to avoid translating a disfluent word, a new edge which skips the word is introduced into the lattice when the word has a higher P_d than a threshold θ. During decoding the importance of this newly introduced edge is optimized by weights based on the disfluency proba-

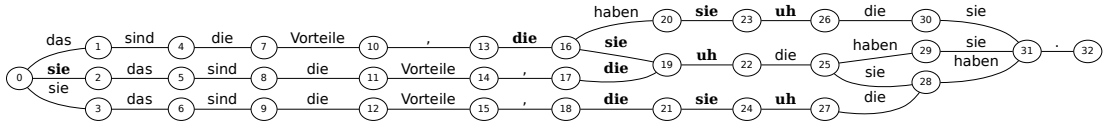


Figure 1: Reordering lattice before adding alternative clean paths for an exemplary sentence

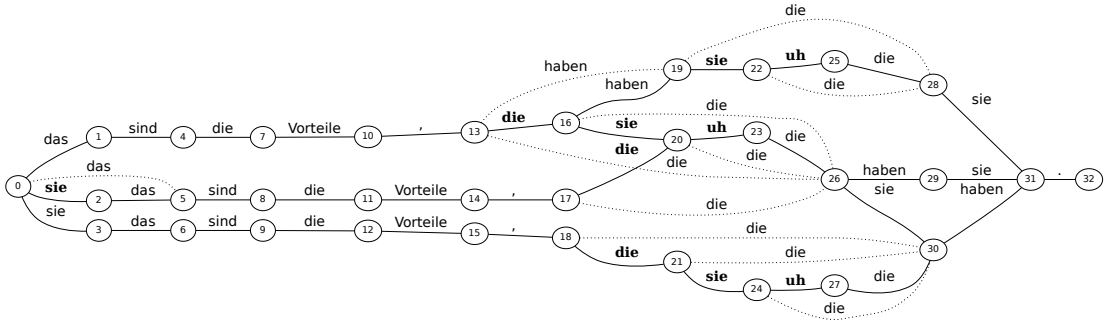


Figure 2: Extended lattice with alternative clean paths for an exemplary sentence

das	0.000732	**sie**	**0.953126**
sind	0.004445	**uh**	**0.999579**
die	0.013451	die	0.029010
Vorteile	0.008183	sie	0.001426
,	0.035408	haben	0.000108
die	**0.651642**	.	0.000033

Table 1: Disfluency probability of each word

bility and transition probability. The extended lattice for the given sentence with $\theta = 0.5$ is shown in Figure 2, with alternative paths marked by a dotted line. The optimal value of θ was manually tuned on the development set.

4 System Description

The training data for our MT system consists of 1.76 million sentences of German-English parallel data. Parallel TED talks[1] are used as in-domain data and our translation models are adapted to the domain. Before training, we apply preprocessing such as text normalization, tokenization, and smartcasing. Additionally, German compound words are split.

To build the phrase table we use the Moses package (Koehn et al., 2007). An LM is trained on 462 million words in English using the SRILM Toolkit (Stolcke, 2002). In order to extend source word context, we use a bilingual LM (Niehues et al., 2011). We use an in-house decoder (Vogel, 2003) with minimum error rate training (Venugopal et al., 2005) for optimization.

For training and testing the CRF model, we use 61k annotated words of manual transcripts of university lectures in German. For tuning and testing the MT system, the same data is used along with its English reference translation. In order to make the best use of the data, we split it into three parts and perform three-fold cross validation. Therefore, the train/development data consists of around 40k words, or 2k sentences, while the test data consists of around 20k words, or 1k sentences.

[1] http://www.ted.com

5 Experiments

In order to compare the effect of the tight integration with other disfluency removal strategies, we conduct different experiments on manual transcripts as well as on the ASR output.

5.1 Manual Transcripts

As a baseline for manual transcripts, we use the whole uncleaned data for development and test. For "No *uh*", we remove the obvious filler words *uh* and *uhm* manually. In the CRF-hard experiment, the token is removed if the label output of the CRF model is a disfluency class. The fourth experiment uses the tight integration scheme, where new source paths which jump over the potentially noisy words are inserted based on the disfluency probabilities assigned by the CRF model. In the next experiments, this method is combined with other aforementioned approaches. First, we apply the tight integration scheme after we remove all obvious filler words. In the next experiment, we first remove all words whose P_d is higher than 0.9 as early pruning and then apply the tight integration scheme. In a final experiment, we conduct an oracle experiment, where all words annotated as a disfluency are removed.

5.2 ASR Output

The same experiments are applied to the ASR output. Since the ASR output does not contain reliable punctuation marks, there is a mismatch between the training data of the CRF model, which is manual transcripts with all punctuation marks, and the test data. Thus, we insert punctuation marks and augment sentence boundaries in the ASR output using the monolingual translation system (Cho et al., 2012). As the sentence boundaries differ from the reference translation, we use the Levenshtein minimum edit distance algorithm (Matusov et al., 2005) to align hypothesis for evaluation. No optimization is conducted, but the scaling factors obtained when using the correponding setup of manual transcripts are used for testing.

5.3 Results

Table 2 shows the results of our experiments. The scores are reported in case-sensitive BLEU (Papineni et al., 2002).

System	Dev	Text	ASR
Baseline	23.45	22.70	14.50
No *uh*	25.09	24.04	15.10
CRF-hard	25.32	24.50	15.15
Tight int.	25.30	24.59	15.19
No *uh* + Tight int.	25.41	24.68	15.33
Pruning + Tight int.	25.38	**24.84**	**15.51**
Oracle	25.57	24.87	-

Table 2: Translation results for the investigated disfluency removal strategies

Compared to the baseline where all disfluencies are kept, the translation quality is improved by 1.34 BLEU points for manual transcripts by simply removing all obvious filler words. When we take the output of the CRF as a hard decision, the performance is further improved by 0.46 BLEU points. When using the tight integration scheme, we improve the translation quality around 0.1 BLEU points compared to the CRF-hard decision. The performance is further improved by removing *uh* and *uhm* before applying the tight integration scheme. Finally the best score is achieved by using the early pruning coupled with the tight integration scheme. The translation score is 0.34 BLEU points higher than the CRF-hard decision. This score is only 0.03 BLEU points less than the oracle case, without all disfluencies.

Experiments on the ASR output also showed a considerable improvement despite word errors and consequently decreased accuracy of the CRF detection. Compared to using only the CRF-hard decision, using the coupled approach improved the performance by 0.36 BLEU points, which is 1.0 BLEU point higher than the baseline.

System	Precision	Recall
CRF-hard	0.898	0.544
Pruning + Tight int.	0.937	0.521

Table 3: Detection performance comparison

Table 3 shows a comparison of the disfluency detection performance on word tokens. While recall is slightly worse for the coupled approach, precision is improved by 4% over the hard decision, indicating that the tight integration scheme decides more accurately. Since deletions made by a hard decision can not be recovered and losing a meaningful word on the source side can be very critical, we believe that precision is more important for this task. Consequently we retain more words on the source side with the tight integration scheme, but the numbers of word tokens on the translated target side are similar. The translation model is able to leave out unnecessary words during translation.

6 Conclusion

We presented a novel scheme to integrate disfluency removal into the MT process. Using this scheme, it is possible to consider disfluency probabilities during decoding and therefore to choose words which can lead to better translation performance. The disfluency probability of each token is obtained from a CRF model, and is encoded in the word lattice. Additional edges are added in the word lattice, to bypass the words potentially representing speech disfluencies.

We achieve the best performance using the tight integration method coupled with early pruning. This method yields an improvement of 2.1 BLEU points for manual transcripts and 1.0 BLEU point improvement over the baseline for ASR output.

Although the translation of ASR output is improved using the suggested scheme, there is still room to improve. In future work, we would like to improve performance of disfluency detection for ASR output by including acoustic features in the model.

Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 287658.

References

Eunah Cho, Jan Niehues, and Alex Waibel. 2012. Segmentation and Punctuation Prediction in Speech Language Translation using a Monolingual Translation System. In *Proceedings of the International Workshop for Spoken Language Translation (IWSLT)*, Hong Kong, China.

Eunah Cho, Thanh-Le Ha, and Alex Waibel. 2013. CRF-based Disfluency Detection using Semantic Features for German to English Spoken Language Translation. In *Proceedings of the International Workshop for Spoken Language Translation (IWSLT)*, Heidelberg, Germany.

Chris Dyer. 2009. Using a Maximum Entropy Model to Build Segmentation Lattices for MT. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Boulder, Colorado, USA, June. Association for Computational Linguistics.

Erin Fitzgerald, Kieth Hall, and Frederick Jelinek. 2009. Reconstructing False Start Errors in Spontaneous Speech Text. In *Proceedings of the European Association for Computational Linguistics (EACL)*, Athens, Greece.

Teresa Herrmann, Jan Niehues, and Alex Waibel. 2013. Combining Word Reordering Methods on different Linguistic Abstraction Levels for Statistical Machine Translation. In *Proceedings of the Seventh Workshop on Syntax, Semantics and Structure in Statistical Translation*, Altanta, Georgia, USA, June. Association for Computational Linguistics.

Matthias Honal and Tanja Schultz. 2003. Correction of Disfluencies in Spontaneous Speech using a Noisy-Channel Approach. In *Eurospeech*, Geneva.

Mark Johnson and Eugene Charniak. 2004. A TAG-based Noisy Channel Model of Speech Repairs. In *Proceedings of the Association for Computational Linguistics (ACL)*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Association for Computational Linguistics (ACL), Demonstration Session*, Prague, Czech Republic, June.

Taku Kudoh. 2007. CRF++: Yet Another CRF Toolkit.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional Random Fields: Probabilitic Models for Segmenting and Labeling Sequence Data. In *ICML*, Massachusetts, USA.

Sameer Maskey, Bowen Zhou, and Yuqing Gao. 2006. A Phrase-Level Machine Translation Approach for Disfluency Detection using Weighted Finite State Tranducers. In *Interspeech*, Pittsburgh, PA.

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Herrmann Ney. 2005. Evaluating Machine Translation Output with Automatic Sentence Segmentation. In *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)*, Boulder, Colorado, USA, October.

Jan Niehues and Muntsin Kolss. 2009. A POS-Based Model for Long-Range Reorderings in SMT. In *Proceedings of the 4th Workshop on Statistical Machine Translation*, Athens, Greece.

Jan Niehues, Teresa Herrmann, Stephan Vogel, and Alex Waibel. 2011. Wider Context by Using Bilingual Language Models in Machine Translation. In *Proceedings of the 6th Workshop on Statistical Machine Translation*, Edinburgh, UK.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. Technical Report RC22176 (W0109-022), IBM Research Division, T. J. Watson Research Center.

Kay Rottmann and Stephan Vogel. 2007. Word Reordering in Statistical Machine Translation with a POS-Based Distortion Model. In *TMI*, Skövde, Sweden.

Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. Denver, Colorado, USA.

Ashish Venugopal, Andreas Zollman, and Alex Waibel. 2005. Training and Evaluation Error Minimization Rules for Statistical Machine Translation. In *WPT-05*, Ann Arbor, MI.

Stephan Vogel. 2003. SMT Decoder Dissected: Word Reordering. In *Int. Conf. on Natural Language Processing and Knowledge Engineering*, Beijing, China.

Wen Wang, Gokhan Tur, Jing Zheng, and Necip Fazil Ayan. 2010. Automatic Disfluency Removal for Improving Spoken Language Translation. In *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Non-Monotonic Parsing of *Fluent umm I Mean* Disfluent Sentences

Mohammad Sadegh Rasooli
Department of Computer Science
Columbia University, New York, NY, USA
rasooli@cs.columbia.edu

Joel Tetreault
Yahoo Labs
New York, NY, USA
tetreaul@yahoo-inc.com

Abstract

Parsing disfluent sentences is a challenging task which involves detecting disfluencies as well as identifying the syntactic structure of the sentence. While there have been several studies recently into solely detecting disfluencies at a high performance level, there has been relatively little work into joint parsing and disfluency detection that has reached that state-of-the-art performance in disfluency detection. We improve upon recent work in this joint task through the use of novel features and learning cascades to produce a model which performs at 82.6 F-score. It outperforms the previous best in disfluency detection on two different evaluations.

1 Introduction

Disfluencies in speech occur for several reasons: hesitations, unintentional mistakes or problems in recalling a new object (Arnold et al., 2003; Merlo and Mansur, 2004). Disfluencies are often decomposed into three types: filled pauses (IJ) such as "uh" or "huh", discourse markers (DM) such as "you know" and "I mean" and edited words (reparandum) which are repeated or corrected by the speaker (repair). The following sentence illustrates the three types:

I want a flight $\underbrace{\text{to Boston}}_{\text{Reparandum}}$ $\underbrace{\text{uh}}_{\text{IJ}}$ $\underbrace{\text{I mean}}_{\text{DM}}$ $\underbrace{\text{to Denver}}_{\text{Repair}}$

To date, there have been many studies on disfluency detection (Hough and Purver, 2013; Rasooli and Tetreault, 2013; Qian and Liu, 2013; Wang et al., 2013) such as those based on TAGs and the noisy channel model (e.g. Johnson and Charniak (2004), Zhang et al. (2006), Georgila (2009), and Zwarts and Johnson (2011)). High performance disfluency detection methods can greatly enhance the linguistic processing pipeline of a spoken dialogue system by first "cleaning" the speaker's utterance, making it easier for a parser to process correctly. A joint parsing and disfluency detection model can also speed up processing by merging the disfluency and parsing steps into one. However, joint parsing and disfluency detection models, such as Lease and Johnson (2006), based on these approaches have only achieved moderate performance in the disfluency detection task. Our aim in this paper is to show that a high performance joint approach is viable.

We build on our previous work (Rasooli and Tetreault, 2013) (henceforth RT13) to jointly detect disfluencies while producing dependency parses. While this model produces parses at a very high accuracy, it does not perform as well as the state-of-the-art in disfluency detection (Qian and Liu, 2013) (henceforth QL13). In this paper, we extend RT13 in two important ways: 1) we show that by adding a set of novel features selected specifically for disfluency detection we can outperform the current state of the art in disfluency detection in two evaluations[1] and 2) we show that by extending the architecture from two to six classifiers, we can drastically increase the speed and reduce the memory usage of the model without a loss in performance.

2 Non-monotonic Disfluency Parsing

In transition-based dependency parsing, a syntactic tree is constructed by a set of stack and buffer actions where the parser greedily selects an action at each step until it reaches the end of the sentence with an empty buffer and stack (Nivre, 2008). A state in a transition-based system has a stack of words, a buffer of unprocessed words and a set of arcs that have been produced in the parser history. The parser consists of a state (or a configuration)

[1] Honnibal and Johnson (2014) have a forthcoming paper based on a similar idea but with a higher performance.

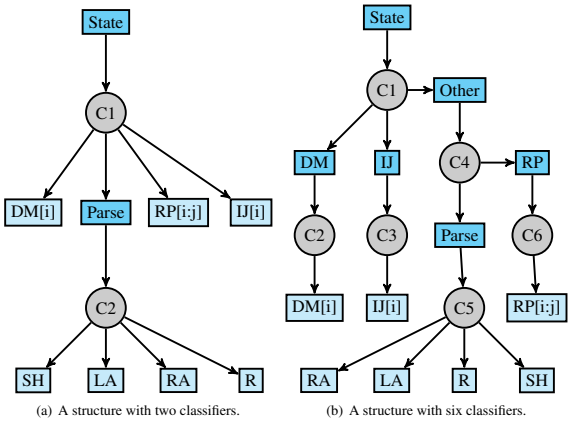
Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 48–53,
Gothenburg, Sweden, April 26-30 2014.

which is manipulated by a set of actions. When an action is made, the parser goes to a new state.

The arc-eager algorithm (Nivre, 2004) is a transition-based algorithm for dependency parsing. In the initial state of the algorithm, the buffer contains all words in the order in which they appear in the sentence and the stack contains the artificial *root* token. The actions in arc-eager parsing are left-arc (LA), right-arc (RA), reduce (R) and shift (SH). LA removes the top word in the stack by making it the dependent of the first word in the buffer; RA shifts the first word in the buffer to the stack by making it the dependent of the top stack word; R pops the top stack word and SH pushes the first buffer word into the stack.

The arc-eager algorithm is a monotonic parsing algorithm, i.e. *once an action is performed, subsequent actions should be consistent with it* (Honnibal et al., 2013). In monotonic parsing, if a word becomes a dependent of another word or acquires a dependent, other actions shall not change those dependencies that have been constructed for that word in the action history. Disfluency removal is an issue for monotonic parsing in that if an action creates a dependency relation, the other actions cannot repair that dependency relation. The main idea proposed by RT13 is to change the original arc-eager algorithm to a non-monotonic one so it is possible to repair a dependency tree while detecting disfluencies by incorporating three new actions (one for each disfluency type) into a two-tiered classification process. The structure is shown in Figure 1(a). In short, at each state the parser first decides between the three new actions and a parse action (C1). If the latter is selected, another classifier (C2) is used to select the best parse action as in normal arc eager parsing.

The three additional actions to the arc-eager algorithm to facilitate disfluency detection are as follows: **1) RP[i:j]**: From the words outside the buffer, remove words i to j from the sentence and tag them as *reparandum*, delete all of their dependencies and push all of their dependents onto the stack. **2) IJ[i]**: Remove the first i words from the buffer (without adding any dependencies to them) and tag them as *interjection*. **3) DM[i]**: Remove the first i words from the buffer (without adding any dependencies) and tag them as *discourse marker*.



(a) A structure with two classifiers. (b) A structure with six classifiers.

Figure 1: Two kinds of cascades for disfluency learning. Circles are classifiers and light-colored blocks show the final decision by the system.

3 Model Improvements

To improve upon RT13, we first tried to learn all actions jointly. Essentially, we added the three new actions to the original arc-eager action set. However, this method (henceforth M1) performed poorly on the disfluency detection task. We believe this stems from a feature mismatch, i.e. some of the features, such as rough copies, are only useful for reparanda while some others are useful for other actions. Speed is an additional issue. Since for each state, there are many candidates for each of the actions, the space of possible candidates makes the parsing time potentially squared.

Learning Cascades One possible solution for reducing the complexity of the inference is to formulate and develop learning cascades where each cascade is in charge of a subset of predictions with its specific features. For this task, it is not essential to always search for all possible phrases because only a minority of cases in speech texts are disfluent (Bortfeld et al., 2001). For addressing this problem, we propose M6, a new structure for learning cascades, shown in Figure 1(b) with a more complex structure while more efficient in terms of speed and memory. In the new structure, we do not always search for all possible phrases which will lead to an expected linear time complexity. The main processing overhead here is the number of decisions to make by classifiers but this is not as time-intensive as finding all candidate phrases in all states.

Feature Templates RT13 use different feature sets for the two classifiers: C2 uses the parse fea-

tures promoted in Zhang and Nivre (2011, Table 1) and C1 uses features which are shown with regular font in Figure 2. We show that one can improve RT13 by adding new features to the C1 classifier which are more appropriate for detecting reparanda (shown in bold in Figure 2). We call this new model M2E, "E" for extended. In Figure 3, the features for each classifier in RT13, M2E, M6 and M1 are described.

We introduce the following new features: **LIC** looks at the number of common words between the reparandum candidate and words in the buffer; e.g. if the candidate is "to Boston" and the words in the buffer are "to Denver", LIC[1] is one and LIC[2] is also one. In other words, LIC is an indicator of a rough copy. The **GPNG** (post n-gram feature) allows us to model the fluency of the resulting sentence after an action is performed, without explicitly going into it. It is the count of possible n-grams around the buffer after performing the action; e.g. if the candidate is a reparandum action, this feature introduces the n-grams which will appear after this action. For example, if the sentence is "I want a flight to Boston | to Denver" (where | is the buffer boundary) and the candidate is "to Boston" as reparandum, the sentence will look like "I want a flight | to Denver" and then we can count all possible n-grams (both lexicalized and unlexicalized) in the range i and j inside and outside the buffer. **GBPF** is a collection of baseline parse features from (Zhang and Nivre, 2011, Table 1).

The need for classifier specific features becomes more apparent in the M6 model. Each of the classifiers uses a different set of features to optimize performance. For example, LIC features are only useful for the sixth classifier while post n-gram features are useful for C2, C3 and C6. For the joint model we use the C1 features from M2B and the C1 features from M6.

Abbr.	Description
GS[i/j]	First n Ws/POS outside β (n=1:i/j)
GB[i/j]	First n Ws/POS inside β (n=1:i/j)
GL[i/j]	Are n Ws/POS i/o β equal? (n=1:i/j)
GT[i]	n last FGT; e.g. *parse:la* (n=1:i)
GTP[i]	n last FGT e.g. *parse* (n=1:i)
GGT[i]	n last FGT + POS of β_0 (n=1:i)
GGTP[i]	n last CGT + POS of β_0 (n=1:i)
GN[i]	(n+m)-gram of m/n POS i/o β (n,m=1:i)
GIC[i]	# common Ws i/o β (n=1:i)
GNR[i]	Rf. (n+m)-gram of m/n POS i/o β (n,m=1:i)
GPNG[i/j]	PNGs from n/m Ws/POS i/o β (m,n:1:i/j)
GBPF	Parse features (Zhang and Nivre, 2011)
LN[i,j]	First n Ws/POS of the cand. (n=1:i/j)
LD	Distance between the cand. and s_0
LL[i,j]	first n Ws/POS of rp and β equal? (n=1:i/j)
LIC[i]	# common Ws for rp/repair (n=1:i)

Figure 2: Feature templates used in this paper and their abbreviations. β: buffer, β_0: first word in the buffer, s_0: top stack word, Ws: words, rp: reparadnum, cand.: candidate phrase, PNGs: post n-grams, FGT: fine-grained transitions and CGT: coarse-grained transitions. *Rf. n-gram*: n-gram from unremoved words in the state.

Classifier	Features
M2 Features	
C1 (RT13)	GS[4/4], GB[4/4], GL[4/6], GT[5], GTP[5] GGT[5], GGTP[5], GN[4], GNR[4], GIC[6] LL[4/6], LD
C1 (M2E)	RT13 $\cup$ (LIC[6], GBPF, GPNG[4/4]) - LD
C2	GBPF
M6 Features	
C1	GBPF, GB[4/4], GL[4/6], GT[5], GTP[5] GGT[5], GGTP[5], GN[4], GNR[4], GIC[6]
C2	GB[4/4], GT[5], GTP[5], GGT[5], GGTP[5] GN[4], GNR[4], GPNG[4/4], LD, LN[24/24]
C3	GB[4/4], GT[5], GTP[5], GGT[5], GGTP[5] GN[4], GNR[4], GPNG[4/4], LD, LN[12/12]
C4	GBPF, GS[4/6], GT[5], GTP[5], GGT[5] GGTP[5], GN[4], GNR[4], GIC[13]
C5	GBPF
C6	GBPF, LL[4/6], GPNG[4/4] LN[6/6], LD, LIC[13]
M1 Features: RT13 C1 features $\cup$ C2 features	

Figure 3: Features for each model. M2E is the same as RT13 with extended features (bold features in Figure 2). M6 is the structure with six classifiers. Other abbreviations are described in Figure 2.

4 Experiments and Evaluation

We evaluate our new models, M2E and M6, against prior work on two different test conditions. In the first evaluation (Eval 1), we use the parsed section of the Switchboard corpus (Godfrey et al., 1992) with the train/dev/test splits from Johnson and Charniak (2004) (JC04). All experimental settings are the same as RT13. We compare our new models against this prior work in terms of disfluency detection performance and parsing accuracy. In the second evaluation (Eval 2), we compare our work against the current best disfluency detection method (QL13) on the JC04 split as well as on a 10 fold cross-validation of the parsed section of the Switchboard. We use gold POS tags for all evaluations.

For all of the joint parsing models we use the weighted averaged Perceptron which is the same as averaged Perceptron (Collins, 2002) but with a

loss weight of two for reparandum candidates as done in prior work. The standard arc-eager parser is first trained on a "cleaned" Switchboard corpus (i.e. after removing disfluent words) with 3 training iterations. Next, it is updated by training it on the real corpus with 3 additional iterations. For the other classifiers, we use the same number of iterations determined from the development set.

Eval 1 The disfluency detection and parse results on the test set are shown in Table 1 for the four systems (M1, RT13, M2E and M6). The joint model performs poorly on the disfluency detection task, with an F-score of 41.5, and the prior work performance which serves as our baseline (RT13) has a performance of 81.4. The extended version of this model (M2E) raises performance substantially to 82.2. This shows the utility of training the C1 classifier with additional features. Finally, the M6 classifier is the top performing model at 82.6.

	Disfluency			Parse	
Model	Pr.	Rec.	F1	UAS	F1
M1	27.4	**85.8**	41.5	60.2	64.6
RT13	85.1	77.9	81.4	88.1	87.6
M2E	**88.1**	77.0	82.2	88.1	87.6
M6	87.7	78.1	**82.6**	**88.4**	**87.7**

Table 1: Comparison of joint parsing and disfluency detection methods. UAS is the unlabeled parse accuracy score.

The upperbound for the parser attachment accuracy (UAS) is 90.2 which basically means that if we have gold standard disfluencies and remove disfluent words from the sentence and then parse the sentence with a regular parser, the UAS will be 90.2. If we had used the regular parser to parse the disfluent sentences, the UAS for correct words would be 70.7. As seen in Table 1, the best parser UAS is 88.4 (M6) which is very close to the upperbound, however RT13, M2E and M6 are nearly indistinguishable in terms of parser performance.

Eval 2 To compare against QL13, we use the second version of the publicly provided code and modify it so it uses gold POS tags and retrain and optimize it for the parsed section of the Switchboard corpus (these are known as *mrg* files, and are a subset of the section of the Switchboard corpus used in QL13, known as *dps* files). Since their system has parameters tuned for the *dps* Switchboard corpus we retrained it for a fair comparison. As in the reimplementation of RT13, we have evaluated the QL13 system with optimal number of training iterations (10 iterations). As seen in Table 2, although the annotation in the *mrg* files is less precise than in the *dps* files, M6 outperforms all models on the JC04 split thus showing the power of the new features and new classifier structure.

Model	JC04 split	xval
RT13	81.4	81.6
QL13 (optimized)	82.5	82.2
M2E	82.2	**82.8**
M6	**82.6**	82.7

Table 2: Disfluency detection results (F1 score) on JC04 split and with cross-validation (xval)

To test for robustness of our model, we perform 10-fold cross validation after clustering files based on their name alphabetic order and creating 10 data splits. As seen in Table 2, the top model is actually M2E, nudging out M6 by 0.1. More noticeable is the difference in performance over QL13 which is now 0.6.

Speed and memory usage Based on our Java implementation on a 64-bit 3GHz Intel CPU with 68GB of memory, the speed for M6 (36 ms/sent) is 3.5 times faster than M2E (128 ms/sent) and 5.2 times faster than M1 (184 ms/sent) and it requires half of the nonzero features overall compared to M2E and one-ninth compared to M1.

5 Conclusion and Future Directions

In this paper, we build on our prior work by introducing rich and novel features to better handle the detection of reparandum and by introducing an improved classifier structure to decrease the uncertainty in decision-making and to improve parser speed and accuracy. We could use early updating (Collins and Roark, 2004) for learning the greedy parser which is shown to be useful in greedy parsing (Huang and Sagae, 2010). K-beam parsing is a way to improve the model though at the expense of speed. The main problem with k-beam parsers is that it is complicated to combine classifier scores from different classifiers. One possible solution is to modify the three actions to work on just one word per action, thus the system will run in completely linear time with one classifier and k-beam parsing can be done by choosing better features for the joint parser. A model similar to this idea is designed by Honnibal and Johnson (2014).

Acknowledgement We would like to thank the reviewers for their comments and useful insights. The bulk of this research was conducted while both authors were working at Nuance Communication, Inc.'s Laboratory for Natural Language Understanding in Sunnyvale, CA.

References

Jennifer E. Arnold, Maria Fagnano, and Michael K. Tanenhaus. 2003. Disfluencies signal theee, um, new information. *Journal of Psycholinguistic Research*, 32(1):25–36.

Heather Bortfeld, Silvia D. Leon, Jonathan E. Bloom, Michael F. Schober, and Susan E. Brennan. 2001. Disfluency rates in conversation: Effects of age, relationship, topic, role, and gender. *Language and Speech*, 44(2):123–147.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 111–118, Barcelona, Spain. Association for Computational Linguistics.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, pages 1–8. Association for Computational Linguistics.

Kallirroi Georgila. 2009. Using integer linear programming for detecting speech disfluencies. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 109–112, Boulder, Colorado. Association for Computational Linguistics.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-92)*, volume 1, pages 517–520.

Matthew Honnibal and Mark Johnson. 2014. Joint incremental disuency detection and dependency parsing. *Transactions of the Association for Computational Linguistics (TACL)*, to appear.

Matthew Honnibal, Yoav Goldberg, and Mark Johnson. 2013. A non-monotonic arc-eager transition system for dependency parsing. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 163–172, Sofia, Bulgaria. Association for Computational Linguistics.

Julian Hough and Matthew Purver. 2013. Modelling expectation in the self-repair processing of annotat-, um, listeners. In *The 17th Workshop on the Semantics and Pragmatics of Dialogue*.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1077–1086, Uppsala, Sweden. Association for Computational Linguistics.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy channel model of speech repairs. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL'04), Main Volume*, pages 33–39, Barcelona, Spain.

Matthew Lease and Mark Johnson. 2006. Early deletion of fillers in processing conversational speech. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 73–76, New York City, USA. Association for Computational Linguistics.

Sandra Merlo and Letıcia Lessa Mansur. 2004. Descriptive discourse: topic familiarity and disfluencies. *Journal of Communication Disorders*, 37(6):489–503.

Joakim Nivre. 2004. Incrementality in deterministic dependency parsing. In *Proceedings of the Workshop on Incremental Parsing: Bringing Engineering and Cognition Together*, pages 50–57. Association for Computational Linguistics.

Joakim Nivre. 2008. Algorithms for deterministic incremental dependency parsing. *Computational Linguistics*, 34(4):513–553.

Xian Qian and Yang Liu. 2013. Disfluency detection using multi-step stacked learning. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 820–825, Atlanta, Georgia. Association for Computational Linguistics.

Mohammad Sadegh Rasooli and Joel Tetreault. 2013. Joint parsing and disfluency detection in linear time. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 124–129, Seattle, Washington, USA. Association for Computational Linguistics.

Wen Wang, Andreas Stolcke, Jiahong Yuan, and Mark Liberman. 2013. A cross-language study on automatic speech disfluency detection. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 703–708, Atlanta, Georgia. Association for Computational Linguistics.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 188–193, Portland, Oregon, USA. Association for Computational Linguistics.

Qi Zhang, Fuliang Weng, and Zhe Feng. 2006. A progressive feature selection algorithm for ultra large feature spaces. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 561–568, Sydney, Australia. Association for Computational Linguistics.

Simon Zwarts and Mark Johnson. 2011. The impact of language models and loss functions on repair disfluency detection. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 703–711, Portland, Oregon, USA. Association for Computational Linguistics.

Lightly-Supervised Word Sense Translation Error Detection for an Interactive Conversational Spoken Language Translation System

Dennis N. Mehay, Sankaranarayanan Ananthakrishnan and Sanjika Hewavitharana
Speech, Language and Multimedia Processing Unit
Raytheon BBN Technologies
Cambridge, MA, 02138, USA
{dmehay,sanantha,shewavit}@bbn.com

Abstract

Lexical ambiguity can lead to concept transfer failure in conversational spoken language translation (CSLT) systems. This paper presents a novel, classification-based approach to accurately detecting word sense translation errors (WSTEs) of ambiguous source words. The approach requires minimal human annotation effort, and can be easily scaled to new language pairs and domains, with only a word-aligned parallel corpus and a small set of manual translation judgments. We show that this approach is highly precise in detecting WSTEs, even in highly skewed data, making it practical for use in an interactive CSLT system.

1 Introduction

Lexical ambiguity arises when a single word form can refer to different concepts. Selecting a contextually incorrect translation of such a word — here referred to as a *word sense translation error* (WSTE) — can lead to a critical failure in a conversational spoken language translation (CSLT) system, where accuracy of concept transfer is paramount. Interactive CSLT systems are especially prone to mis-translating less frequent word senses, when they use phrase-based statistical machine translation (SMT), due to its limited use of source context (source phrases) when constructing translation hypotheses. Figure 1 illustrates a typical WSTE in a phrase-based English-to-Iraqi Arabic CSLT system, where the English word *board*

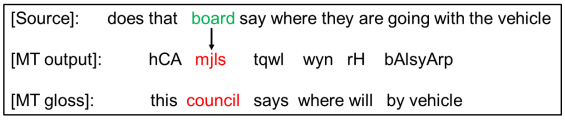


Figure 1: Example WSTE in English-to-Iraqi SMT.

is mis-translated as *mjls* ("*council*"), completely distorting the intended message.

Interactive CSLT systems can mitigate this problem by automatically detecting WSTEs in SMT hypotheses, and engaging the operator in a clarification dialogue (e.g. requesting an unambiguous rephrasing). We propose a novel, two-level classification approach to accurately detect WSTEs. In the first level, a bank of word-specific classifiers predicts, given a rich set of contextual and syntactic features, a distribution over possible target *translations* for each ambiguous source word in our inventory. A single, second-level classifier then compares the predicted target words to those chosen by the decoder and determines the likelihood that an error was made.

A significant novelty of our approach is that the first-level classifiers are fully unsupervised with respect to manual annotation and can easily be expanded to accommodate new ambiguous words and additional parallel data. The other innovative aspect of our solution is the use of a small set of manual translation judgments to train the second-level classifier. This classifier uses high-level features derived from the output of the first-level classifiers to produce a binary WSTE prediction, and can be re-used unchanged even when the first level of classifiers is expanded.

Our goal departs from the large body of work devoted to lightly-supervised word sense disambiguation (WSD) using monolingual and bilingual corpora (Yarowsky, 1995; Schutze, 1998; Diab and Resnik, 2002; Ng et al., 2003; Li and Li, 2002; Purandare and Pedersen, 2004), which seeks to la-

Disclaimer: This paper is based upon work supported by the DARPA BOLT program. The views expressed here are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

Distribution Statement A (Approved for Public Release, Distribution Unlimited)

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 54–58,
Gothenburg, Sweden, April 26-30 2014.

bel and group unlabeled sense instances. Instead, our approach detects *mis-translations* of a known set of ambiguous words.

The proposed method also deviates from existing work on global lexical selection models (Mauser et al., 2009) and on integration of WSD features within SMT systems with the goal of improving offline translation performance (Chan et al., 2007). Rather, we *detect* translation errors due to ambiguous source words with the goal of providing feedback to and soliciting clarification from the system operator in real time. Our approach is partly inspired by Carpuat and Wu's (2007b; 2007a) unsupervised sense disambiguation models for offline SMT. More recently, Carpuat et al. (2013) identify unseen target senses in new domains, but their approach requires the full test corpus upfront, which is unavailable in spontaneous CSLT. Our approach can, in principle, identify novel senses when unfamiliar source contexts are encountered, but this is not our current focus.

2 Baseline SMT System

In this paper, we focus on WSTE detection in the context of phrase-based English-to-Iraqi Arabic SMT, an integral component of our interactive, two-way CSLT system that mediates conversation between monolingual speakers of English and Iraqi Arabic. The parallel training corpus of approximately 773K sentence pairs (7.3M English words) was derived from the DARPA TransTac English-Iraqi two-way spoken dialogue collection and spans a variety of domains including force protection, medical diagnosis and aid, etc. Phrase pairs were extracted from bidirectional IBM Model 4 word alignment after applying a merging heuristic similar to that of Koehn et al. (2003). A 4-gram target LM was trained on Iraqi Arabic transcriptions. Our phrase-based decoder, similar to Moses (Koehn et al., 2007), performs beam search stack decoding based on a standard log-linear model, whose parameters were tuned with MERT (Och, 2003) on a held-out development set (3,534 sentence pairs, 45K words). The BLEU and METEOR scores of this system on a separate test set (3,138 sentence pairs, 38K words) were 16.1 and 42.5, respectively.

3 WSTE Detection

The core of the WSTE detector is a novel, two-level classification pipeline. Our approach avoids the need for expensive, sense-labeled training data based on the observation that knowing the *sense* of an ambiguous source word is distinct from knowing whether a *sense translation error* has occurred. Instead, the target (Iraqi Arabic) words typically associated with a given sense of an ambiguous source (English) word serve as implicit sense labels, as the following describes.

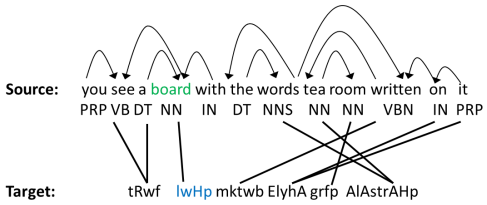


Figure 2: An English–Iraqi training pair.

3.1 A First Level of Unsupervised Classifiers

The main intuition behind our approach is that strong disagreement between the expanded context of an ambiguous source word and the corresponding SMT hypothesis indicates an increased likelihood that a WSTE has occurred. To identify such disagreement, we train a bank of maximum-entropy classifiers (Berger et al., 1996), one for each ambiguous word. The classifiers are trained on the same word-aligned parallel data used for training the baseline SMT system, as follows.

For each instance of an ambiguous source word in the training set, and for each target word it is aligned to, we emit a training instance associating that target word and the wider source context of the ambiguous word. Figure 2 illustrates a typical training instance for the ambiguous English word *board*, which emits a tuple of contextual features and the aligned Iraqi Arabic word *lwHp* ("*placard*") as a target label. We use the following contextual features similar to those of Carpuat and Wu (2005), which are in turn based on the classic WSD features of Yarowsky (1995).

Neighboring Words/Lemmas/POSs. The tokens, t, to the left and right of the current ambiguous token, as well as all trigrams of tokens that span the current token. Separate features for word, lemma and parts of speech tokens, t.

Lemma/POS Dependencies. The lemma-lemma and POS-POS labeled and unlabeled directed syntactic dependencies of the current ambiguous token.

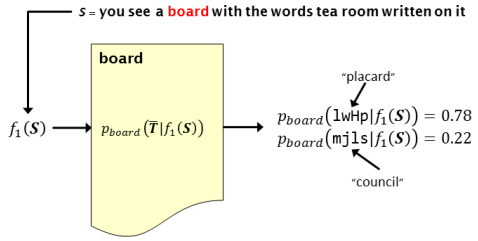


Figure 3: An unsupervised first-level classifier.

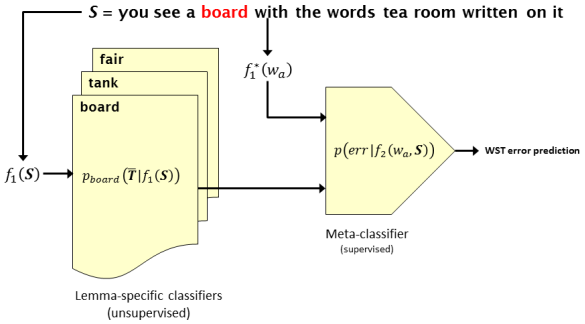


Figure 4: The two-level WSTE architecture.

Bag-of-words/lemmas. Distance decayed bag-of-words-style features for each word and lemma in a seven-word window around the current token.

Figure 3 schematically illustrates how this classifier operates on a sample test sentence. The example assumes that the ambiguous English word *board* is only ever associated with the Iraqi Arabic words *lwHp (“placard”)* and *mjls (“council”)* in the training word alignment. We emphasize that even though the first-level maximum entropy classifiers are intrinsically supervised, their training data is derived via unsupervised word alignment.

3.2 A Second-Level Meta-Classifier

The first-level classifiers do not directly predict the presence of a WSTE, but induce a distribution over possible target words that could be generated by the ambiguous source word in that context. In order to make a binary decision, this distribution must be contrasted with the corresponding target phrase hypothesized by the SMT decoder. One straightforward approach, which we use as a baseline, is to threshold the posterior probability of the word in the SMT target phrase which is ranked highest in the classifier-predicted distribution. However, this approach is not ideal because each classifier has a different target label set and is trained on a different number of instances.

To address this issue, we introduce a second meta-classifier, which is trained on a small number of hand-annotated translation judgments of SMT hypotheses of source sentences containing ambiguous words. The bilingual annotator was simply asked to label the phrasal translation of source phrases containing ambiguous words as *correct* or *incorrect*. We obtained translation judgments for 511 instances from the baseline SMT development and test sets, encompassing 147 pre-defined ambiguous words obtained heuristically from WordNet, public domain homograph lists, etc.

The second-level classifier is trained on a small set of meta-features drawn from the predictions of the first-level classifiers and from simple statistics of the training corpus. For an ambiguous word w_a in source sentence S, with contextual features $f_1(S)$, and aligned to target words $t \in T$ (the set of words in the target phrase) in the SMT hypothesis, we extract the following features:

1. The first-level classifier’s maximum likelihood of any decoded target word: $\max_{t \in T} p_{w_a}(t|f_1(S))$
2. The entropy of the predicted distribution: $\sum_t p_{w_a}(t|f_1(S)) \cdot ln(p_{w_a}(t|f_1(S)))$
3. The number of training instances for w_a
4. The inverse of the number of distinct target labels for w_a.
5. The product of meta-features (1) and (4)

A high value for feature 1 indicates that the first-level model and the SMT decoder agree. By contrast, a high value for feature 2 indicates uncertainty in the classifier’s prediction, due either to a novel source context, or inadequate training data. Feature 3 indicates whether the second scenario of meta-feature 2 might be at play, and feature 4 can be thought of as a simple, uniform prior for each classifier. Finally, feature 5 attenuates feature 1 by this simple, uniform prior. We feed these features to a random forest (Breiman, 2001), which is a committee of decision trees, trained using randomly selected features and data points, using the implementation in Weka (Hall et al., 2009). The target labels for training the second-level classifier are obtained from the binary translation judgments on the small annotated corpus. Figure 4 illustrates the interaction of the two levels of classification.

3.3 Scalability and Portability

Scalability was an important consideration in designing the proposed WSTE approach. For instance, we may wish to augment the inventory with new ambiguous words if the vocabulary grows due to addition of new parallel data or due to a change in the domain. The primary advantage of the two-level approach is that new ambiguous words can be accommodated by augmenting the unsupervised first-level classifier set with additional word-specific classifiers, which can be done by simply extending the pre-defined list of ambiguous words. Further, the current classification stack requires only ≈1.5GB of RAM and performs per-word WSTE inference in only a few milliseconds on a commodity, quad-core laptop, which is critical for real-time, interactive CSLT.

The minimal annotation requirements also allow a high level of *portability* to new language pairs. Moreover, as our results indicate (below), a good quality WSTE detector can be bootstrapped for a new language pair *without any annotation effort* by simply leveraging the first-level classifiers.

4 Experimental Results

The 511 WSTE-annotated instances used for training the second-level classifier doubled as an evaluation set using the leave-one-out cross-validation method. Of these, 115 were labeled as errors by the bilingual judge, while the remaining 396 were translated correctly by the baseline SMT system. The error prediction score from the second-level classifier was thresholded to obtain the receiver operating characteristic (ROC) curve shown in the top (black) curve of Figure 5. We obtain a 43% error detection rate with only 10% false alarms and 71% detection with 20% false alarms, in spite of the highly skewed label distribution. In absolute terms, true positives outnumber false alarms at both the 10% (49 to 39) and 20% (81 to 79) false alarm rates. This is important for deployment, as we do not want to disrupt the flow of conversation with more false alarms than true positives.

For comparison, the bottom (red) ROC curve shows the performance of a baseline WSTE predictor comprised of just meta-feature (1), obtainable directly from the first-level classifiers. This performs slightly worse than the two-level model at 10% false alarms (40% detection, 46 true positives, 39 false alarms), and considerably worse at 20% false alarms (57% detection, 66 true positives, 78 false alarms). Nevertheless, this result indicates the possibility of bootstrapping a good quality baseline WSTE detector in a new language or domain without any annotation effort.

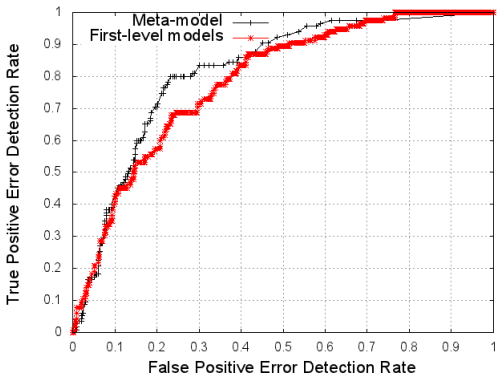


Figure 5: WST error detection ROC curve.

5 Conclusion

We proposed a novel, lightly-supervised, two-level classification architecture that identifies possible mis-translations of pre-defined ambiguous source words. The WSTE detector pre-empts communication failure in an interactive CSLT system by serving as a trigger for initiating feedback and clarification. The first level of our detector comprises of a bank of word-specific classifiers trained on automatic word alignment over the SMT parallel training corpus. Their predicted distributions over target words feed into the second-level meta-classifier, which is trained on a small set of manual translation judgments. On a 511-instance test set, the two-level approach exhibits WSTE detection rates of 43% and 71% at 10% and 20% false alarm rates, respectively, in spite of a nearly 1:4 skew against actual WSTE instances.

Because adding new ambiguous words to the inventory only requires augmenting the set of first-level unsupervised classifiers, our WSTE detection approach is *scalable* to new domains and training data. It is also easily *portable* to new language pairs due to the minimal annotation effort required for training the second-level classifier. Finally, we show that it is possible to bootstrap a good quality WSTE detector in a new language pair *without any annotation effort* using only unsupervised classifiers and a parallel corpus.

References

Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71.

Leo Breiman. 2001. Random Forests. Technical report, Statistics Department, University of California, Berkeley, Berkeley, CA, USA, January.

Marine Carpuat and Dekai Wu. 2005. Word sense disambiguation vs. statistical machine translation. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 387–394.

Marine Carpuat and Dekai Wu. 2007a. How phrase sense disambiguation outperforms word sense disambiguation for statistical machine translation. In *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2007)*, Skovde, Sweden, September.

Marine Carpuat and Dekai Wu. 2007b. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic, June.

Marine Carpuat, Hal Daumé III, Katharine Henry, Ann Irvine, Jagadeesh Jagarlamudi, and Rachel Rudinger. 2013. Sensespotting: Never let your parallel data tie you to an old domain. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1435–1445, Sofia, Bulgaria, August. Association for Computational Linguistics.

Yee Seng Chan, Hwee Tou Ng, and David Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 33–40, Prague, Czech Republic, June.

Mona Diab and Philip Resnik. 2002. An unsupervised method for word sense tagging using parallel corpora. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 255–262, July.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA Data Mining Software: An Update. *SIGKDD Explorations*, 11(1).

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *NAACL '03: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54, Morristown, NJ, USA. Association for Computational Linguistics.

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

Hang Li and Cong Li. 2002. Word translation disambiguation using bilingual bootstrapping. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 343–351, July.

Arne Mauser, Saša Hasan, and Hermann Ney. 2009. Extending statistical machine translation with discriminative and trigger-based lexicon models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 210–218, Singapore, August. Association for Computational Linguistics.

Hwee Tou Ng, Bin Wang, and Yee Seng Chan. 2003. Exploiting parallel texts for word sense disambiguation: An empirical study. In *Proceedings of 41st Annual Meeting on Association for Computational Linguistics*, pages 455–462, July.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *ACL '03: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 160–167, Morristown, NJ, USA. Association for Computational Linguistics.

Amruta Purandare and Ted Pedersen. 2004. Word sense discrimination by clustering contexts in vector and similarity spaces. In *Proceedings of the Conference on Computational Natural Language Learning*, pages 41–48.

Hinrich Schutze. 1998. Automatic word sense discrimination. *Journal of Computational Linguistics*, 24:97–123.

David Yarowsky. 1995. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, ACL '95, pages 189–196, Stroudsburg, PA, USA. Association for Computational Linguistics.

Map Translation Using Geo-tagged Social Media

Sunyou Lee, Taesung Lee, Seung-won Hwang
POSTECH, Korea
{sylque,elca4u,swhwang}@postech.edu

Abstract

This paper discusses the problem of map translation, of servicing spatial entities in multiple languages. Existing work on entity translation harvests translation evidence from text resources, not considering spatial locality in translation. In contrast, we mine geo-tagged sources for multilingual tags to improve recall, and consider spatial properties of tags for translation to improve precision. Our approach empirically improves accuracy from 0.562 to 0.746 using Taiwanese spatial entities.

1 Introduction

A map is becoming an essential online service for mobile devices, providing a current location and generating directions to spatial entities (SEs). Although major map services aim to support a map in more than 100 local languages, their current support is often biased either to English or local maps. For example, Figure 1 contrasts richly populated Taiwanese entities (in the local language) whereas only some of those entities are translated in English version. Our goal is to translate richly populated SEs into another language, in the finer granularity such as restaurants.

A baseline approach would be adopting existing work on entity transliteration work, which uses phonetic similarity, such as translating 'Barack Obama' into '贝拉克·奥巴马' [Beilake·Aobama]. Another approach is using automatically-harvested or manually-built translation resources, such as multilingual Gazetteer (or, SE dictionary[1]). However, these resources are often limited to well-known or large SEs, which leads to translation with near-perfect precision but low recall.

[1]For example, http://tgnis.ascc.net provides SE translation pairs.

Moreover, blindly applying existing entity translation methods to SE translation leads to extremely low accuracy. For example, an SE '十分車站' should be translated into 'Shifen station', where '十分' is transliterated to [Shifen], whereas '車站' is semantically translated based on its meaning 'station'. However, due to this complex nature often observed in SE translation, an off-the-shelf translation service (e.g., Google Translate) returns 'very station'[2] as an output. In addition, SE names are frequently abbreviated so that we cannot infer the meanings to semantically translate them. For instance, 'United Nations' is often abbreviated into 'UN' and its translation is also often abbreviated. As a result, the abbreviation in the two languages, (UN, 联合国), shares neither phonetic nor semantic similarity.

To overcome these limitations, we propose to extract and leverage properties of SEs from a social media, namely Flickr. Especially, we exploit co-occurrence of names in two different languages. For example, '台北' co-occurs with its English translation 'Taipei' as tags on the same photo. This is strong evidence that they are translations of each other. In addition to co-occurrence, we leverage spatial properties of SEs. For example, among tags that frequently co-occur with '台北', such as 'Taipei' and 'Canon', 'Taipei' is

[2]As of Dec 26, 2013.

Figure 1: A map of Taipei in English. Google Maps, as of Oct 14, 2013

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 59–63,
Gothenburg, Sweden, April 26-30 2014.

Symbols	Description
$\mathbb{C}$	A set of all Chinese spatial entities
c	A Chinese spatial entity, $c \in \mathbb{C}$
e	An English entity
p	A photo
D	Photos
D_c	Photos with c
D_e	Photos with e
E_c	a set of English tags from D_c
G_c	a set of GPS coordinates from D_c
G_e	a set of GPS coordinates from D_e

Table 1: Overview of symbols

more likely to be its correct translation because the spatial distributions of the two tags are similarly skewed in the same area. Our approach significantly improves the F1-score (0.562 to 0.746), compared to an off-the-shelf translators.

2 Overall Framework

We provide the framework of our proposed method using predefined symbols (Table 1). We consider a scenario of translating each SE c in a set of all SEs $\mathbb{C}$ in a Chinese map into English so that we obtain an English map[3].

STEP 1. Finding a set D_c: We crawl a photo set D with tags from Flickr. We consider each of the tags as an entity. Given an SE $c \in \mathbb{C}$, we find a set $D_c \subseteq D$. For each photo in D_c, we obtain a set of tags in multiple languages and GPS coordinates of the photo as translation evidence (Table 2).

STEP 2. Collecting candidate English tags: To obtain translation candidates of c, we build a set E_c of English tags that co-occur with c, and a set $D_e \subseteq D$ of photos for each $e \in E_c$.

STEP 3. Calculating matching score $w(c, e)$: For an English candidate $e \in E_c$, we calculate the matching score between c and e, and translate c into e with the highest $w(c, e)$ score. We describe the details of computing $w(c, e)$ in Section 3.

[3]We use an example of translating from Chinese to English for illustration, but we stress that our work straightforwardly extends if multilingual tags of these two languages are sufficient.

Photos	Chinese tag	English tag
p_1	女王頭	Taipei, The Queen's Head, food
p_2	愛河	love river, food, park, dog
p_3	野柳, 女王頭	Yehliu, Taipei, food
p_4	台北, 東北角, 女王頭	The Queen's Head, Taipei, restaurant
p_5	淡水河	Taipei, Tamsui river, dog, food

Table 2: Structure of crawled photos $D = \{p_1, p_2, p_3, p_4, p_5\}$

e	The Queen's Head	Taipei
D_e	$\{p_1, p_4\}$	$\{p_1, p_3, p_4, p_5\}$
$CF(c, e)$ **(FB)**	2	3
$TS(c, e)$	0	-0.3
$w(c, e)$ **(SB)**	0	-0.9

Table 3: **SB** vs. **FB**: Translating c = 女王頭 into $e \in E_{\text{女王頭}}$ where $D_{\text{女王頭}} = \{p_1, p_3, p_4\}$

3 Matching Score

3.1 Naive Approach: Frequency-based Translation (FB)

A naive solution for map translation is to use co-occurrence of multilingual tags. For example, if a Chinese tag '女王頭' frequently co-occurs with an English tag 'The Queen's Head', we can translate '女王頭' into 'The Queen's Head'. Specifically, for a given Chinese SE c and a candidate English tag e, we define *co-occurring frequency* $CF(c, e)$.

Definition. *Co-occurring Frequency* $CF(c, e)$. Co-occurring frequency $CF(c, e)$ is the number of photos in which c and e are co-tagged,

$$CF(c, e) = |D_c \cap D_e|, \quad (1)$$

where D_c and D_e are photos with a Chinese SE c and an English tag e, respectively.

We compute $CF(c, e)$ for all candidates in $e \in E_c$ and rank them. Then, **FB** translates c into e with the highest $CF(c, e)$ score. However, **FB** cannot address the following two challenges that occur due to tag sparseness.

- C1 : Large regions such as 'Taiwan', 'Taipei' (Section 3.2)
- C2 : Non-SEs such as 'dog', 'food' (Section 3.3)

3.2 Overcoming C1: Scarcity-biased Translation (SB)

Users tend to tag photos with both a specific SE and large administrative regions such as 'Taiwan' and 'Taipei', which makes **FB** score of large regions higher than the proper one. For example, 'Taipei' is tagged in most photos in D (Table 2); therefore, $CF(\text{女王頭}, \text{Taipei})$ larger than $CF(\text{女王頭}, \text{The Queen's Head})$ (Table 3).

To reduce the effect of large regions, we introduce a new feature to give high scores for specific SEs (e.g., 'The Queen's Head'). We observe that a large region's tag is associated with many photos in $D - D_c$, whereas a scarce but useful tag is particularly tagged in D_c. We consider $\frac{|D_e|}{|D-D_c|}$ to measure how many photos have e without c. Therefore, $\frac{|D_e|}{|D-D_c|}$ increases as e frequently appears where c does not. In contrast, if e appears mostly with c, the ratio decreases. Taking inverse of the ratio to give higher score when e appears mostly with c, we define *tag scarcity* $TS(c, e)$ and apply it to the candidate ranking function.

Definition. *Tag scarcity* $TS(c, e)$. Given an SE c and a candidate English tag $e \in E_c$, the tag scarcity is defined as

$$TS(c, e) = \log |D - D_c|/|D_e|. \quad (2)$$

Definition. *Scarcity-biased Matching Score* $w(c, e)$. Given an SE c and a candidate English tag $e \in E_c$, the matching score between c and e is

$$w(c, e) = CF(c, e) \times TS(c, e). \quad (3)$$

To illustrate the effect of **SB** with our running example (Table 2), we compare 'The Queen's Head' to 'Taipei' for translating '女王頭' (Table 3). **FB** gives a higher score to 'Taipei' than to the correct translation 'The Queen's Head'. In contrast, by reflecting TS, **SB** correctly concludes that 'The Queen's Head' is the best match.

Apart from **SB**, we can also leverage an additional resource such as an *administrative hierarchy*, if exists, to blacklist some large regions' names from E_c. By first translating larger regions and excluding them, the precision for translating small SEs can increase. For instance, we translate a country '台灣 (Taiwan)' earlier than a city '台北 (Taipei)'. Then, when translating '台北', even though $CF(\text{台北}, \text{Taiwan})$ is higher than $CF(\text{台北}, \text{Taipei})$, we ignore 'Taiwan' in $E_{\text{台北}}$ because it is already matched with '台灣'.

3.3 Overcoming C2: Pruning Non-SEs (PN)

We prune non-SEs such as 'food' based on spatial locality of a tag. We observe that the GPS coordinates G_e of photos with an SE tag e tend to be more concentrated in a specific region than those of photos with a non-SE. For instance, comparing a non-SE 'food' and an SE 'The Queen's Head', the GPS coordinates in G_{food} are more widespread all over Taiwan than those in $G_{\text{The Queen's Head}}$.

We leverage the coordinates of a *distant SE pair*. For example, two spatially far SEs '台北 (Taipei)' and '台東 (Taitung)' compose a distant SE pair. Because both SEs are unlikely to be tagged in a single photo, an English tag that co-occurs with both of them would be a non-SE.

Formally, we define two Chinese SEs c_1 and c_2 as a distant SE pair if $G_{c_1} \cap G_{c_2} = \emptyset$, and M as a set of all distant SE pairs among $\mathbb{C} \times \mathbb{C}$. We judge that an English tag e is a non-SE if G_e intersects with both G_{c_1} and G_{c_2} for a distant pair c_1 and c_2. Formally, an English tag e is non-SE if the following equation $PN(e)$ is nonzero.

$$PN(e) = \sum_{(c_1,c_2)\in M} |G_{c_1} \cap G_e| \times |G_{c_2} \cap G_e|. \quad (4)$$

4 Evaluation

4.1 Experimental Setting

Photo Data and Ground Truth: We crawled 227,669 photos taken in Taipei from Flickr, which also provided GPS coordinates of photos. We took a set D of 148,141 photos containing both Chinese and English tags and manually labelled 200 gold standard Chinese-English SE pairs whose names appeared together in at least one photo in D.

Administrative Hierarchy: An administrative hierarchy was obtained from *Taiwan Geographical Names Information System* [4].

Baselines: We chose baselines available for many languages except for the gazetteer and excluded methods that used specific textual corpora.

- Phonetic Similarity (PH) (Kim et al., 2013)
- Off-the-shelf Translator: Google Translate[5], Bing Translator[6]
- Taiwanese-English Gazetteer (official SE translation[4])

[4] http://tgnis.ascc.net/. Its latest modification has done on August 23, 2013.

[5] http://translate.google.co.kr/

[6] http://www.bing.com/translator

Chinese SE [Transliteration]	SB+PN	PH	Google Translate	Bing Translator	Gazetteer
兔子餐廳 [Tuzi Canting]	**To House**	Astrid	Rabbit Restaurant	Hare House	ø
典華旗艦館 [Dianhua Gijianguan]	**Denwell Restaurant**	Taipei Restaurants	Dianhua Flagship Museum	Classic China Flagship Center	ø

Table 4: Example translation from our method and the baselines (Correct translations are boldfaced.)

Method	P	R	F1
Transliteration	.463	.463	.463
Google Translate	.562	**.562**	.562
Bing Translator	.425	.425	.425
Taiwanese-English Gazetteer	**.960**	.485	**.645**

Table 5: P, R, and F1 of baselines

Method	P	R	F1	MRR
FB	.215	.215	.215	.439
FB + PN	.220	.220	.220	.454
SB	.640	.640	.640	.730
SB + PN	**.680**	**. 670**	**.675**	**.752**

(a) Without administrative hierarchy

Method	P	R	F1	MRR
FB	.515	.515	.515	.641
FB + PN	.624	.615	.620	.730
SB	.655	.655	.655	.733
SB + PN	.706	.695	.700	.763
FB + SB + PN	**.751**	**.740**	**.746**	**.806**

(b) With given hierarchy

Table 6: Effect of FB, SB, PN, and the hierarchy

Measures: We measured *precision* (P), *recall* (R), *F1-Score* (F1), and *mean reciprocal rank* (MRR) where MRR $= \frac{1}{|P|}\sum_{(c,e_0)\in P}\frac{1}{rank(c,e_0)}$, for which P is a set of gold standard pairs (c, e_0) of a Chinese SE c and its correct translation e_0, and $rank(c, e_0)$ indicates the rank of $w(c, e_0)$ among all $w(c, e)$ s.t. $e \in E_c$.

4.2 Experimental Results

Comparison to Baselines: The proposed approach (**SB + PN**) with or without the administrative hierarchy provided higher R and F1 than did the baseline methods (Table 5, 6).

The baseline methods showed generally low P, R, and F1. Especially, the gazetteer produced high precision, but poor recall because it could not translate lesser-known SEs such as '兔子餐廳 (To House)' and '典華旗艦館 (Denwell Restaurant)' (Table 4).

Effect of SB and PN: We experimented on the effect of the combinations of the features (Table 6). Using all the features **FB+SB+PN** with hierarchy, which translated the upper level of the hierarchy with **FB** and the lower level with **SB**, showed the best effectiveness. Simple **FB** gave both low precision and very low recall regardless of whether we used the hierarchy. Replacing **FB** with **SB** yielded both higher F1 and higher MRR.

PN increased F1, especially greatly when it was used with **SB** or the hierarchy because **PN** filtered out different types of noises, non-SEs. Applying **PN**, we classified 361 non-SEs and 6 SEs as noises in total. Despite some misclassifications, it improved the overall accuracy by ignoring highly ranked non-SEs such as 'dog' and 'food'.

5 Conclusion

We propose a scalable map translator that uses a geo-tagged corpus from social media to mine translation evidence to translate between English and maps in local languages. Our approach leverages both co-occurrence of the SE tags in Chinese and English and their scarcity and spatial property. Our approach can translate small or emerging spatial entities such as restaurants, which major map services cannot support currently. We empirically validated that our approach provided higher P, R, F1, and MRR than the existing methods including popular off-the-shelf translation services.

Acknowledgments

This research was supported by the MSIP (The Ministry of Science, ICT and Future Planning), Korea and Microsoft Research, under IT/SW Creative research program supervised by the NIPA(National IT Industry Promotion Agency). (NIPA-2013-H0503-13-1009).

References

Jinhan Kim, Seung-won Hwang, Long Jiang, Y Song, and Ming Zhou. 2013. Entity translation mining from comparable corpora: Combining graph mapping with corpus latent features. *IEEE Transactions on Knowledge and Data Engineering*, 25(8):1787–1800.

Predicting Romanian Stress Assignment

Alina Maria Ciobanu[1,2], Anca Dinu[1,3], Liviu P. Dinu[1,2]
[1] Center for Computational Linguistics, University of Bucharest
[2] Faculty of Mathematics and Computer Science, University of Bucharest
[3] Faculty of Foreign Languages and Literatures, University of Bucharest
alina.ciobanu@my.fmi.unibuc.ro, anca_d_dinu@yahoo.com, ldinu@fmi.unibuc.ro

Abstract

We train and evaluate two models for Romanian stress prediction: a baseline model which employs the consonant-vowel structure of the words and a cascaded model with averaged perceptron training consisting of two sequential models – one for predicting syllable boundaries and another one for predicting stress placement. We show in this paper that Romanian stress is predictable, though not deterministic, by using data-driven machine learning techniques.

1 Introduction

Romanian is a highly inflected language with a rich morphology. As dictionaries usually fail to cover the pronunciation aspects for all word forms in languages with such a rich and irregular morphology (Sef et al., 2002), we believe that a data-driven approach is very suitable for syllabication and stress prediction for Romanian words. Moreover, such a system proves extremely useful for inferring syllabication and stress placement for out-of-vocabulary words, for instance neologisms or words which recently entered the language.

Even if they are closely related, Romanian stress and syllabication were unevenly studied in the computational linguistic literature, i.e., the Romanian syllable received much more attention than the Romanian stress (Dinu and Dinu, 2005; Dinu, 2003; Dinu et al., 2013; Toma et al., 2009). One possible explanation for the fact that Romanian syllabication was more intensively studied than Romanian stress is the immediate application of syllabication to text editors which need reliable hyphenation. Another explanation could be that most linguists (most recently Dindelegan (2013)) insisted that Romanian stress is not predictable, thus discouraging attempts to investigate any systematic patterns.

Romanian is indeed a challenging case study, because of the obvious complexities of the data with respect to stress assignment. At first sight, no obvious patterns emerge for learning stress placement (Dindelegan, 2013), other than as part of individual lexical items. The first author who challenges this view is Chitoran (2002), who argues in favor of the predictability of the Romanian stress system. She states that stress placement strongly depends on the morphology of the language, more precisely on the distribution of the lexical items based on their part of speech (Chitoran, 1996). Thus, considering this type of information, lexical items can be clustered in a limited number of regular subpatterns and the unpredictability of stress placement is significantly reduced. A rule-based method for lexical stress prediction on Romanian was introduced by Oancea and Badulescu (2002).

Dou et al. (2009) address lexical stress prediction as a sequence tagging problem, which proves to be an accurate approach for this task. The effectiveness of using conditional random fields for orthographic syllabication is investigated by Trogkanis and Elkan (2010), who employ them for determining syllable boundaries and show that they outperform previous methods. Bartlett et al. (2008) use a discriminative tagger for automatic orthographic syllabication and present several approaches for assigning labels, including the language-independent *Numbered NB* tag scheme, which labels each letter with a value equal to the distance between the letter and the last syllable boundary. According to Damper et al. (1999), syllable structure and stress pattern are very useful in text-to-speech synthesis, as they provide valuable knowledge regarding the pronunciation modeling. Besides converting the letters to the corresponding phonemes, information about syllable boundaries and stress placement is also needed for the correct synthesizing of a word in grapheme-to-phoneme conversion (Demberg et al., 2007).

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 64–68,
Gothenburg, Sweden, April 26-30 2014.

In this paper, we rely on the assumption that the stress system of Romanian is predictable. We propose a system for automatic prediction of stress placement and we investigate its performance by accounting for several fine-grained characteristics of Romanian words: part of speech, number of syllables and consecutive vowels. We investigate the consonant-vowel structure of the words (C/V structure) and we detect a high number of stress patterns. This calls for the need of machine learning techniques, in order to automatically learn such a wide range of variational patterns.

2 Approach

We address the task of stress prediction for Romanian words (out-of-context) as a sequence tagging problem. In this paper, we account only for the primary stress, but this approach allows further development in order to account for secondary stress as well. We propose a cascaded model consisting of two sequential models trained separately, the output of the first being used as input for the second. We use averaged perceptron for parameter estimation and three types of features which are described in detail further in this section: n-grams of characters, n-grams marking the C/V structure of the word and binary positional indicators of the current character with respect to the syllable structure of the word. We use one sequential model to predict syllable boundaries and another one to predict stress placement. Previous work on orthographic syllabication for Romanian (Dinu et al., 2013) shows that, although a rule-based algorithm models complex interactions between features, its practicality is limited. The authors report experiments on a Romanian dataset, where the rule-based algorithm is outperformed by an SVM classifier and a CRF system with character n-gram features.

We use a simple tagging structure for marking primary stress. The stressed vowel receives the positive tag 1, while all previous characters are tagged 0 and all subsequent ones 2. This structure helps enforce the uniqueness of the positive tag. The main features used are character n-grams up to $n = W$ in a window of radius W around the current position. For example, if $W = 2$, the feature template consists of `c[-2]`, `c[-1]`, `c[0]`, `c[1]`, `c[2]`, `c[-2:-1]`, `c[-1:0]`, `c[0:1]`, `c[1:2]`. If the current letter is the fourth of the word *dinosaur*, *o*, the feature values would be *i, n, o, s, a, in, no, os, sa*. We use two additional types of features:

- features regarding the C/V structure of the word: n-grams using, instead of characters, markers for consonants (C) and vowels (V);
- binary indicators of the following positional statements about the current character, related to the statistics reported in Table 1:
  - exactly before/after a split;
  - in the first/second/third/fourth syllable of the word, counting from left to right;
  - in the first/second/third/fourth syllable of the word, counting from right to left

The syllabication prediction is performed with another sequential model of length $n-1$, where each node corresponds to a position between two characters. Based on experimenting and previous work, we adopted the *Numbered NB* labeling. Each position is labeled with an integer denoting the distance from the previous boundary. For example, for the word *diamond*, the syllable (above) and stress annotations (below) are as follows:

```
d   i   a   m   o   n   d
  1   0   0   1   2   3
0   1   2   2   2   2   2
```

The features used for syllabication are based on the same principle, but because the positions are in-between characters, the window of radius W has length $2W$ instead of $2W+1$. For this model we used only character n-grams as features.

3 Data

We run our experiments for Romanian using the *RoSyllabiDict* (Barbu, 2008) dictionary, which is a dataset of annotated words comprising 525,528 inflected forms for approximately 65,000 lemmas. This is, to our best knowledge, the largest experiment conducted and reported for Romanian so far. For each entry, the syllabication and the stressed vowel (and, in case of ambiguities, also grammatical information or type of syllabication) are provided. For example, the word *copii* (*children*) has the following representation:

```
<form w="copii" obs="s."> co-píi</form>
```

We investigate stress placement with regard to the syllable structure and we provide in Table 1 the percentages of words having the stress placed on different positions, counting syllables from the beginning and from the end of the words as well.

For our experiments, we discard words which do not have the stressed vowel marked, compound

Syllable	%words
1st	5.59
2nd	18.91
3rd	39.23
4th	23.68
5th	8.52

(a) counting syllables from the beginning of the word

Syllable	%words
1st	28.16
2nd	43.93
3rd	24.14
4th	3.08
5th	0.24

(b) counting syllables from the end of the word

Table 1: Stress placement for *RoSyllabiDict*

words having more than one stressed vowel and ambiguous words (either regarding their part of speech or type of syllabication).

We investigate the C/V structure of the words in *RoSyllabiDict* using raw data, i.e., *a, ă, â, e, i, î, o, u* are always considered vowels and the rest of the letters in the Romanian alphabet are considered consonants. Thus, we identify a very large number of C/V structures, most of which are not deterministic with regard to stress assignment, having more then one choice for placing the stress[1].

4 Experiments and Results

In this section we present the main results drawn from our research on Romanian stress assignment.

4.1 Experiments

We train and evaluate a cascaded model consisting of two sequential models trained separately, the output of the first being used as input to the second. We split the dataset in two subsets: train set (on which we perform cross-validation to select optimal parameters for our model) and test set (with unseen words, on which we evaluate the performance of our system). We use the same train/test sets for the two sequential models, but they are trained independently. The output of the first model (used for predicting syllabication) is used for determining feature values for the second one (used for predicting stress placement) for the test set. The second model is trained using gold syllabication (provided in the dataset) and we report results on the test set in both versions: using gold syllabication to determine feature values and using predicted syllabication to determine feature values. The results with gold syllabication are reported only for providing an upper bound for learning and for comparison.

We use averaged perceptron training (Collins, 2002) from *CRFsuite* (Okazaki, 2007). For the stress prediction model we optimize hyperparameters using grid search to maximize the 3-fold cross-validation F_1 score of class 1, which marks the stressed vowels. We searched over $\{2,3,4\}$ for W and over $\{1,5,10,25,50\}$ for the maximum number of iterations. The values which optimize the system are 4 for W and 50 for the maximum number of iterations. We investigate, during grid search, whether employing C/V markers and binary positional indicators improve our system's performance. It turns out that in most cases they do. For the syllabication model, the optimal hyperparameters are 4 for the window radius and 50 for the maximum number of iterations. We evaluate the cross-validation F_1 score of class 0, which marks the position of a hyphen. The system obtains 0.995 instance accuracy for predicting syllable boundaries.

We use a "majority class" type of baseline which employs the C/V structures described in Section 3 and assigns, for a word in the test set, the stress pattern which is most common in the training set for the C/V structure of the word, or places the stress randomly on a vowel if the C/V structure is not found in the training set[2]. The performance of both models on *RoSyllabiDict* dataset is reported in Table 2. We report word-level accuracy, that is, we account for words for which the stress pattern was correctly assigned. As expected, the cascaded model performs significantly better than the baseline.

Model	Accuracy
Baseline	0.637
Cascaded model (gold)	0.975
Cascaded model (predicted)	0.973

Table 2: Accuracy for stress prediction

Further, we perform an in-depth analysis of the sequential model's performance by accounting for

[1] For example, for CCV-CVC structure (1,390 occurrences in our dataset) there are 2 associated stress patterns: CCV-CV̲C (1,017 occurrences) and CC̲V-CVC (373 occurrences). Words with 6 syllables cover the highest number of distinct C/V structures (5,749). There are 31 C/V structures (ranging from 4 to 7 syllables) reaching the maximum number of distinct associated stress patterns (6).

[2] For example, the word *copii* (meaning *children*) has the following C/V structure: CV-CVV. In our training set, there are 659 words with this structure and the three stress patterns which occur in the training set are as follows: CV-CV̲V (309 occurrences), C̲V-CVV (283 occurrences) and CV-CV̲V̲ (67 occurrences). Therefore, the most common stress pattern CV-CV̲V is correctly assigned, in this case, for the word *copii*.

several fine-grained characteristics of the words in *RoSyllabiDict*. We divide words in categories based on the following criteria:

- part of speech: verbs, nouns, adjectives
- number of syllables: 2-8, 9+
- number of consecutive vowels: with at least 2 consecutive vowels, without consecutive vowels

Category	Subcategory	♯ words	Accuracy G	Accuracy P
POS	Verbs	167,193	0.995	0.991
	Nouns	266,987	0.979	0.979
	Adjectives	97,169	0.992	0.992
Syllables	2 syllables	34,810	0.921	0.920
	3 syllables	111,330	0.944	0.941
	4 syllables	154,341	0.966	0.964
	5 syllables	120,288	0.981	0.969
	6 syllables	54,918	0.985	0.985
	7 syllables	17,852	0.981	0.989
	8 syllables	5,278	0.992	0.984
	9+ syllables	1,468	0.979	0.980
Vowels	With VV	134,895	0.972	0.972
	Without VV	365,412	0.976	0.974

Table 3: Accuracy for cascaded model with gold (G) and predicted (P) syllabication

We train and test the cascaded model independently for each subcategory in the same manner as we did for the entire dataset. We decided to use cross-validation for parameter selection instead of splitting the data in train/dev/test subsets in order to have consistency across all models, because some of these word categories do not comprise enough words for splitting in three subsets (words with more than 8 syllables, for example, have only 1,468 instances). The evaluation of the system's performance and the number of words in each category are presented in Table 3.

4.2 Results Analysis

The overall accuracy is 0.975 for the cascaded model with gold syllabication and 0.973 for the cascaded model with predicted syllabication. The former system outperforms the latter by only very little. With regard to the part of speech, the highest accuracy when gold syllabication is used was obtained for verbs (0.995), followed by adjectives (0.992) and by nouns (0.979). When dividing the dataset with respect to the words' part of speech, the cascaded model with predicted syllabication is outperformed only for verbs. With only a few exceptions, the accuracy steadily increases with the number of syllables. The peak is reached for words with 6 syllables when using the gold syllabication and for words with 7 syllables when using the predicted syllabication. Although, intuitively, the accuracy should be inversely proportional to the number of syllables, because the number of potential positions for stress placement increases, there are numerous stress patterns for words with 6, 7 or more syllables, which never occur in the dataset[3]. It is interesting to notice that stress prediction accuracy is almost equal for words containing two or more consecutive vowels and for words without consecutive vowels. As expected, when words are divided in categories based on their characteristics the system is able to predict stress placement with higher accuracy.

5 Conclusion and Future Work

In this paper we showed that Romanian stress is predictable, though not deterministic, by using data-driven machine learning techniques. Syllable structure is important and helps the task of stress prediction. The cascaded sequential model using gold syllabication outperforms systems with predicted syllabication by only very little.

In our future work we intend to experiment with other features as well, such as syllable n-grams instead of character n-grams, for the sequential model. We plan to conduct a thorough error analysis and to investigate the words for which the systems did not correctly predict the position of the stressed vowels. We intend to further investigate the C/V structures identified in this paper and to analyze the possibility to reduce the number of patterns by considering details of word structure (for example, instead of using raw data, to augment the model with annotations about which letters are actually vowels) and to adapt the learning model to finer-grained linguistic analysis.

Acknowledgements

The authors thank the anonymous reviewers for their helpful comments. The contribution of the authors to this paper is equal. Research supported by a grant of ANRCS, CNCS UEFISCDI, project number PN-II-ID-PCE-2011-3-0959.

[3]For example, for the stress pattern CV-CV-CV-CV-CV-CVCV, which matches 777 words in our dataset, the stress is never placed on the first three syllables.

References

Ana-Maria Barbu. 2008. Romanian Lexical Data Bases: Inflected and Syllabic Forms Dictionaries. In *Proceedings of the 6th International Conference on Language Resources and Evaluation, LREC 2008*, pages 1937–1941.

Susan Bartlett, Grzegorz Kondrak, and Colin Cherry. 2008. Automatic Syllabification with Structured SVMs for Letter-to-Phoneme Conversion. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, ACL-HLT 2008*, pages 568–576.

Ioana Chitoran. 1996. Prominence vs. rhythm: The predictability of stress in Romanian. In *Grammatical theory and Romance languages*, pages 47–58. Karen Zagona.

Ioana Chitoran. 2002. *The phonology of Romanian. A constraint-based approach*. Mouton de Gruyter.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10, EMNLP 2002*, pages 1–8.

Robert I. Damper, Yannick Marchand, M. J. Adamson, and K. Gustafson. 1999. Evaluating the pronunciation component of text-to-speech systems for English: a performance comparison of different approaches. *Computer Speech & Language*, 13(2):155–176.

Vera Demberg, Helmut Schmid, and Gregor Möhler. 2007. Phonological Constraints and Morphological Preprocessing for Grapheme-to-Phoneme Conversion. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, ACL 2007*, pages 96–103.

Gabriela Pană Dindelegan. 2013. *The Grammar of Romanian*. Oxford University Press.

Liviu P. Dinu and Anca Dinu. 2005. A Parallel Approach to Syllabification. In *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing, CICLing 2005*, pages 83–87.

Liviu P. Dinu, Vlad Niculae, and Octavia-Maria Șulea. 2013. Romanian Syllabication Using Machine Learning. In *Proceedings of the 16th International Conference on Text, Speech and Dialogue, TSD 2013*, pages 450–456.

Liviu Petrisor Dinu. 2003. An Approach to Syllables via some Extensions of Marcus Contextual Grammars. *Grammars*, 6(1):1–12.

Qing Dou, Shane Bergsma, Sittichai Jiampojamarn, and Grzegorz Kondrak. 2009. A Ranking Approach to Stress Prediction for Letter-to-Phoneme Conversion. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th IJCNLP of the AFNLP, ACL 2009*, pages 118–126.

Eugeniu Oancea and Adriana Badulescu. 2002. Stressed Syllable Determination for Romanian Words within Speech Synthesis Applications. *International Journal of Speech Technology*, 5(3):237–246.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of Conditional Random Fields (CRFs).

Tomaz Sef, Maja Skrjanc, and Matjaz Gams. 2002. Automatic Lexical Stress Assignment of Unknown Words for Highly Inflected Slovenian Language. In *Proceedings of the 5th International Conference on Text, Speech and Dialogue, TSD 2002*, pages 165–172.

S.-A. Toma, E. Oancea, and D. Munteanu. 2009. Automatic rule-based syllabication for Romanian. In *Proceedings of the 5th Conference on Speech Technology and Human-Computer Dialogue, SPeD 2009*, pages 1–6.

Nikolaos Trogkanis and Charles Elkan. 2010. Conditional Random Fields for Word Hyphenation. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, ACL 2010*, pages 366–374.

Passive-Aggressive Sequence Labeling with Discriminative Post-Editing for Recognising Person Entities in Tweets

Leon Derczynski
University of Sheffield
leon@dcs.shef.ac.uk

Kalina Bontcheva
University of Sheffield
kalina@dcs.shef.ac.uk

Abstract

Recognising entities in social media text is difficult. NER on newswire text is conventionally cast as a sequence labeling problem. This makes implicit assumptions regarding its textual structure. Social media text is rich in disfluency and often has poor or noisy structure, and intuitively does not always satisfy these assumptions. We explore noise-tolerant methods for sequence labeling and apply discriminative post-editing to exceed state-of-the-art performance for person recognition in tweets, reaching an F1 of 84%.

1 Introduction

The language of social media text is unusual and irregular (Baldwin et al., 2013), with misspellings, non-standard capitalisation and jargon, disfluency and fragmentation. Twitter is one of the sources of social media text most challenging for NLP (Eisenstein, 2013; Derczynski et al., 2013).

In particular, traditional approaches to Named Entity Recognition (NER) perform poorly on tweets, especially on person mentions – for example, the default model of a leading system reaches an F1 of less than 0.5 on person entities in a major tweet corpus. This indicates a need for approaches that can cope with the linguistic phenomena apparently common among social media authors, and operate outside of newswire with its comparatively low linguistic diversity.

So, how can we adapt? This paper contributes two techniques. Firstly, it demonstrates that entity recognition using noise-resistant sequence labeling outperforms state-of-the-art Twitter NER, although we find that recall is consistently lower than precision. Secondly, to remedy this, we introduce a method for automatically post-editing the resulting entity annotations by using a discriminative classifier. This improves recall and precision.

2 Background

Named entity recognition is a well-studied problem, especially on newswire and other long-document genres (Nadeau and Sekine, 2007; Ratinov and Roth, 2009). However, experiments show that state-of-the-art NER systems from these genres do not transfer well to social media text.

For example, one of the best performing general-purpose named entity recognisers (hereon referred to as Stanford NER) is based on linear-chain conditional random fields (CRF) (Finkel et al., 2005). The model is trained on newswire data and has a number of optimisations, including distributional similarity measures and sampling for remote dependencies. While excellent on newswire (overall F1 90%), it performs poorly on tweets (overall F1 44%) (Ritter et al., 2011).

Rule-based named entity recognition has performed a little better on tweets. Another general-purpose NER system, ANNIE (Cunningham et al., 2002), reached F1 of 60% over the same data (Derczynski et al., 2013); still a large difference.

These difficulties spurred Twitter-specific NER research, much of which has fallen into two broad classes: semi-supervised CRF, and LDA-based.

Semi-supervised CRF: Liu et al. (2011) compare the performance of a person name dictionary (F1 of 33%) to a CRF-based semi-supervised approach (F1 of 76% on person names), using a dataset of 12 245 tweets. This, however, is based on a proprietary corpus, and cannot be compared to, since the system is also not available.

Another similar approach is TwiNER (Li et al., 2012), which is focused on a single topic stream as opposed to general-purpose NER. This leads to high performance for a topic-sensitive classifier trained to a particular stream. In contrast we present a general-purpose approach. Further, we extract a specific entity class, where TwiNER performs entity chunking and no classification.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 69–73,
Gothenburg, Sweden, April 26-30 2014.

LDA and vocabularies: Ritter et al. (2011)'s T-NER system uses 2,400 labelled tweets, unlabelled data and Linked Data vocabularies (Freebase), as well as co-training. These techniques helped but did not bring person recognition accuracy above the supervised MaxEnt baseline in their experiments. We use this system as our baseline.

3 Experimental Setup

3.1 Corpus

The experiments combine person annotations from three openly-available datasets: Ritter et al. (2011), UMBC (Finin et al., 2010) and MSM2013 (Basave et al., 2013). In line with previous research (Ritter et al., 2011), annotations on @mentions are filtered out. The placeholder tokens in MSM data (i.e. _MENTION_, _HASHTAG_, _URL_) are replaced with @Mention, #hashtag, and http://url/, respectively, to give case and character n-grams more similar to the original values.

The total corpus has 4 285 tweets, around a third the size of that in Liu et al. (2011). This dataset contains 86 352 tokens with 1 741 entity mentions.

Person entity recognition was chosen as it is a challenging entity type. Names of persons popular on Twitter change more frequently than e.g. locations. Person names also tend to have a long tail, not being confined to just public figures. Lastly, although all three corpora cover different entity types, they all have Person annotations.

3.2 Labeling Scheme

Following Li et al. (2009) we used two-class IO labeling, where each token is either in-entity or out-of-entity. In their NER work, this performed better than the alternative BIO format, since data sparsity is reduced. The IO scheme has the disadvantage of being unable to distinguish cases where multiple different entities of the same type follow each other without intervening tokens. This situation is uncommon and does not arise in our dataset.

3.3 Features

The Stanford NER tool was used for feature generation. When required, nominal values were converted to sparse one-hot vectors. Features for modelling context are included (e.g. ngrams, adjoining labels). Our feature sets were:

base: default Stanford NER features, plus the previous and next token and its word shape.[1]

[1]Default plus useClassFeature=true, noMidNGrams=true, maxNGramLeng=6, usePrev=true, useNext=true, usePrevSequences=true, maxLeft=1, useTypeSeqs=true, useTypeSeqs2=true, useTypeSeqs3=true, useTypeySequences=true, wordShape=chris2useLC, useDisjunctive=true, lowercaseNGrams=true, useShapeConjunctions=true

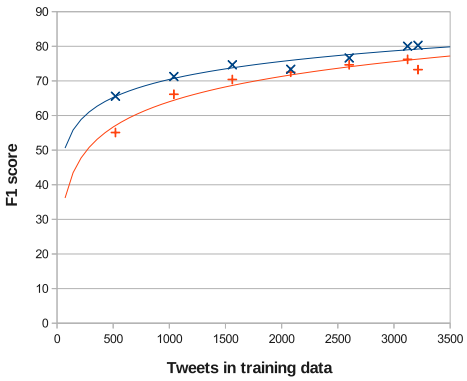


Figure 1: Training curve for *lem*. Diagonal cross (blue) is CRF/PA, vertical cross (red) SVM/UM.

lem: with added lemmas, lower-case versions of tokens, word shape, and neighbouring lemmas (in attempt to reduce feature sparsity & cope better with lexical and orthographic noise). Word shape describes the capitalisation and the type of characters (e.g. letters, numbers, symbols) of a word, without specifying actual character choices. For example, *Capital* may become *Ww*.

These representations are chosen to compare those that work well for newswire to those with scope for tolerance of noise, prevalent in Twitter.

3.4 Classifiers

For structured sequence labeling, we experiment with conditional random fields – CRF (Lafferty et al., 2001) – using the CRFsuite implementation (Okazaki, 2007) and LBFGS. We also use an implementation of the passive-aggressive CRF from CRFsuite, choosing *max_iterations* = *500*.

Passive-aggressive learning (Crammer et al., 2006) demonstrates tolerance to noise in training data, and can be readily adapted to provide structured output, e.g. when used in combination with CRF. Briefly, it skips updates (is *passive*) when the hinge loss of a new weight vector during update is zero, but when it is positive, it aggressively adjusts the weight vector regardless of the required step size. This is integrated into CRF using a damped loss function and passive-aggressive (PA) decisions to choose when to update. We explore the PA-I variant, where the objective function scales linearly with the slack variable.

Approach	Precision	Recall	F1
Stanford	85.88	50.00	63.20
Ritter	77.23	**80.18**	**78.68**
MaxEnt	**86.92**	59.09	70.35
SVM	77.55	59.16	67.11
SVM/UM	73.26	69.63	71.41
CRF	82.94	62.39	71.21
CRF/PA	80.37	65.57	72.22

Table 1: With base features (base)

Approach	Precision	Recall	F1
Stanford	90.60	60.00	72.19
Ritter	77.23	**80.18**	78.68
MaxEnt	**91.10**	66.33	76.76
SVM	88.22	66.58	75.89
SVM/UM	81.16	74.97	77.94
CRF	89.52	70.52	78.89
CRF/PA	86.85	74.71	**80.32**

Table 2: With shape and lemma features (lem)

For independent discriminative classification, we use SVM, SVM/UM and a maximum entropy classifier (MegaM (Daumé III, 2004)). SVM is provided by the SVMlight (Joachims, 1999) implementation. SVM/UM is an uneven margins SVM model, designed to deal better with imbalanced training data (Li et al., 2009).

3.5 Baselines

The first baseline is the Stanford NER CRF algorithm, the second Ritter's NER algorithm. We adapted the latter to use space tokenisation, to preserve alignment when comparing algorithms. Baselines are trained and evaluated on our dataset.

3.6 Evaluation

Candidate entity labelings are compared using the CoNLL NER evaluation tool (Sang and Meulder, 2003), using precision, recall and F1. Following Ritter, we use 25%/75% splits made at tweet, and not token, level.

4 Results

The base feature set performs relatively poorly on all classifiers, with only MaxEnt beating a baseline on any score (Table 1). However, all achieve a higher F1 score than the default Stanford NER. Of these classifiers, SVM/UM achieved the best precision and CRF/PA – the best F1. This demonstrates that the noise-tolerance adaptations to SVM and CRF (uneven margins and passive-aggressive updates, respectively) did provide improvements over the original algorithms.

Results using the extended features (lem) are shown in Table 2. All classifiers improved, including the baseline Stanford NER system. The SVM/UM and CRF/PA adaptations continued to outperform the vanilla models. With these features, MaxEnt achieved highest precision and CRF variants beat both baselines, with a top F1 of 80.32%. We continue using the *lem* feature set.

Entity length (tokens)	Count
1	610
2	1065
3	51
4	15

Table 3: Distribution of person entity lengths.

5 Discriminative Post-Editing

Precision is higher than recall for most systems, especially the best CRF/PA (Table 2). To improve recall, potential entities are re-examined in *post-editing* (Gadde et al., 2011). Manual post-editing improves machine translation output (Green et al., 2013); we train an automatic editor.

We adopt a gazetteer-based approach to triggering a discriminative editor, which makes decisions about labels after primary classification. The gazetteer consists of the top 200 most common names in English speaking countries. The first names of popular figures over the past two years (e.g. *Helle*, *Barack*, *Scarlett*) are also included. This gives 470 case-sensitive *trigger* terms.

Often the trigger term is just the first in a sequence of tokens that make up the person name. As can be seen from the entity length statistics shown in Table 3, examining up to two tokens covers most (96%) person names in our corpus. Based on this observation, we look ahead just one extra token beyond the trigger term. This gives a token sub-sequence that was marked as out-of-entity by the original NER classifier. Its constituents become *candidate* person name tokens.

Candidates are then labeled using a high-recall classifier. The classifier should be instance-based, since we are not labeling whole sequences. We chose SVM with variable cost (Morik et al., 1999), which can be adjusted to prefer high recall.

To train this classifier, we extract a subset of instances from the current training split as follows. Each trigger term is included. Also, if the trigger term is labeled as an entity, each subsequent in-entity token is also included. Finally, the next out-of-entity token is also included, to give examples of when to stop. For example, these tokens are either in or out of the training set:

Method	Missed entity F1	Overall P	Overall R	Overall F1
No editing - plain CRF/PA	0.00	86.85	74.71	80.32
Naïve: trigger token only	5.82	86.61	78.91	82.58
Naïve: trigger plus one	6.05	81.26	82.08	81.67
SVM editor, $Cost = 0.1$	78.26	87.38	79.16	83.07
SVM editor, $Cost = 0.5$	89.72	87.17	80.30	83.60
SVM editor, $Cost = 1.0$	90.74	87.19	80.43	83.67
SVM editor, $Cost = 1.5$	92.73	87.23	80.69	**83.83**
SVM editor, $Cost = 2.0$	92.73	87.23	80.69	**83.83**

Table 4: Post-editing performance. Higher $Cost$ sacrifices precision for recall.

```
Miley     O        in
Heights   O        out

Miley     PERSON   in
Cyrus     PERSON   in
is        O        in
famous    O        out
```

When post-editing, the window is any trigger term and the following token, regardless of initial label. The features used were exactly the same as with the earlier experiment, using the *lem* set. This is compared with two naïve baselines: always annotating trigger terms as Person, and always annotating trigger terms and the next token as Person.

Results are shown in Table 4. Naïve editing baselines had F1 on missed entities of around 6%, showing that post-editing needs to be intelligent.

At $Cost = 1.5$, recall increased to 80.69, exceeding the Ritter recall of 80.18 (raising $Cost$ beyond 1.5 had no effect). This setup gave good accuracy on previously-missed entities (second column) and improved overall F1 to 83.83. It also gave better precision and recall than the best naïve baseline (trigger-only), and 6% absolute higher precision than trigger plus one. This is a 24.2% reduction in error over the Ritter baseline (F1 78.68), and a 17.84% error reduction compared to the best non-edited system (CRF/PA+*lem*).

6 Error Analysis

We examine two types of classification error: false positives (spurious) and false negatives (missed).

False positives occur most often where non-person entities are mentioned. This occurred with mentions of organisations (*Huff Post*), locations (*Galveston*) and products (*Exodus Porter*). Descriptive titles were also sometimes mis-included in person names (*Millionaire Rob Ford*). Names of persons used in other forms also presented as false positives (e.g. *Marie Claire* – a magazine). Polysemous names (i.e. words that could have other functions, such as a verb) were also mis-resolved (*Mark*). Finally, proper nouns referring to groups were sometimes mis-included (*Haitians*).

Despite these errors, precision almost always remained higher than recall over tweets. We use in-domain training data, and so it is unlikely that this is due to the wrong kinds of person being covered in the training data – as can sometimes be the case when applying tools trained on newswire.

False negatives often occurred around incorrect capitalisation and spelling, with unusual names, with ambiguous tokens and in low-context settings. Both omitted and added capitalisation gave false negatives (*charlie gibson*, or *KANYE WEST*). Spelling errors also led to missed names (*Russel Crowe*). Ambiguous names caused false negatives and false positives; our approach missed *mark* used as a name, and the surname of *Jack Straw*. Unusual names with words typically used for other purposes were also not always correctly recognised (e.g. *the Duck Lady*, or the last two tokens of *Spicy Pickle Jr.*). Finally, names with few or no context words were often missed (*Video: Adele 21.*, and *17-9-2010 Tal al-Mallohi, a 19-*).

7 Conclusion

Finding named entities in social media text, particularly tweets, is harder than in newswire. This paper demonstrated that adapted to handle noisy input is useful in this scenario. We achieved the good results using CRF with passive-aggressive updates. We used representations rich in word shape and contextual features and achieved high precision with moderate recall (65.57–74.71).

To improve recall, we added a post-editing stage which finds candidate person names based on trigger terms and re-labels them using a cost-adjusted SVM. This flexible and re-usable approach lead to a final reduction in error rate of 24.2%, giving performance well above that of comparable systems.

Acknowledgment This work received funding from EU FP7 under grant agreement No. 611233, Pheme. We thank Chris Manning and John Bauer of Stanford University for help with the NER tool.

References

T. Baldwin, P. Cook, M. Lui, A. MacKinlay, and L. Wang. 2013. How noisy social media text, how diffrnt social media sources. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 356–364. ACL.

A. E. C. Basave, A. Varga, M. Rowe, M. Stankovic, and A.-S. Dadzie. 2013. Making Sense of Microposts (# MSM2013) Concept Extraction Challenge. In *Proceedings of the Concept Extraction Challenge at the Workshop on 'Making Sense of Microposts'*, volume 1019. CEUR-WS.

K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. 2006. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, 7:551–585.

H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. 2002. GATE: an Architecture for Development of Robust HLT Applications. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, pages 168–175.

H. Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at `http://pub.hal3.name#daume04cg-bfgs`, implementation available at `http://hal3.name/megam/`, August.

L. Derczynski, D. Maynard, N. Aswani, and K. Bontcheva. 2013. Microblog-Genre Noise and Impact on Semantic Annotation Accuracy. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*. ACM.

J. Eisenstein. 2013. What to do about bad language on the internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 359–369. Association for Computational Linguistics.

T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. 2010. Annotating named entities in Twitter data with crowdsourcing. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 80–88.

J. Finkel, T. Grenager, and C. Manning. 2005. Incorporating non-local information into information extraction systems by Gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics.

P. Gadde, L. Subramaniam, and T. A. Faruquie. 2011. Adapting a WSJ trained part-of-speech tagger to noisy text: preliminary results. In *Proceedings of the 2011 Joint Workshop on Multilingual OCR and Analytics for Noisy Unstructured Text Data*. ACM.

S. Green, J. Heer, and C. D. Manning. 2013. The efficacy of human post-editing for language translation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 439–448. ACM.

T. Joachims. 1999. Svmlight: Support vector machine. *SVM-Light Support Vector Machine http://svmlight.joachims. org/, University of Dortmund*, 19(4).

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, San Francisco: Morgan Kaufmann.

Y. Li, K. Bontcheva, and H. Cunningham. 2009. Adapting SVM for Data Sparseness and Imbalance: A Case Study on Information Extraction. *Natural Language Engineering*, 15(2):241–271.

C. Li, J. Weng, Q. He, Y. Yao, A. Datta, A. Sun, and B.-S. Lee. 2012. Twiner: named entity recognition in targeted twitter stream. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 721–730. ACM.

X. Liu, S. Zhang, F. Wei, and M. Zhou. 2011. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 359–367.

K. Morik, P. Brockhausen, and T. Joachims. 1999. Combining statistical learning with a knowledge-based approach-a case study in intensive care monitoring. In *ICML*, volume 99, pages 268–277.

D. Nadeau and S. Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

N. Okazaki. 2007. CRFsuite: a fast implementation of Conditional Random Fields (CRFs).

L. Ratinov and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, pages 147–155. Association for Computational Linguistics.

A. Ritter, S. Clark, Mausam, and O. Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proc. of Empirical Methods for Natural Language Processing (EMNLP)*, Edinburgh, UK.

E. F. T. K. Sang and F. D. Meulder. 2003. Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In *Proceedings of CoNLL-2003*, pages 142–147. Edmonton, Canada.

Accelerated Estimation of Conditional Random Fields using a Pseudo-Likelihood-inspired Perceptron Variant

Teemu Ruokolainen[a] **Miikka Silfverberg**[b] **Mikko Kurimo**[a] **Krister Lindén**[b]
[a] Department of Signal Processing and Acoustics, Aalto University, firstname.lastname@aalto.fi
[b] Department of Modern Languages, University of Helsinki, firstname.lastname@helsinki.fi

Abstract

We discuss a simple estimation approach for conditional random fields (CRFs). The approach is derived heuristically by defining a variant of the classic perceptron algorithm in spirit of pseudo-likelihood for maximum likelihood estimation. The resulting approximative algorithm has a linear time complexity in the size of the label set and contains a minimal amount of tunable hyper-parameters. Consequently, the algorithm is suitable for learning CRF-based part-of-speech (POS) taggers in presence of large POS label sets. We present experiments on five languages. Despite its heuristic nature, the algorithm provides surprisingly competetive accuracies and running times against reference methods.

1 Introduction

The conditional random field (CRF) model (Lafferty et al., 2001) has been successfully applied to several sequence labeling tasks in natural language processing, including part-of-speech (POS) tagging. In this work, we discuss accelerating the CRF model estimation in presence of a large number of labels, say, hundreds or thousands. Large label sets occur in POS tagging of morphologically rich languages (Erjavec, 2010; Haverinen et al., 2013).

CRF training is most commonly associated with the (conditional) maximum likelihood (ML) criterion employed in the original work of Lafferty et al. (2001). In this work, we focus on an alternative training approach using the averaged perceptron algorithm of Collins (2002). While yielding competitive accuracy (Collins, 2002; Zhang and Clark, 2011), the perceptron algorithm avoids extensive tuning of hyper-parameters and regularization required by the stochastic gradient descent algorithm employed in ML estimation (Vishwanathan et al., 2006). Additionally, while ML and perceptron training share an identical time complexity, the perceptron is in practice faster due to sparser parameter updates.

Despite its simplicity, running the perceptron algorithm can be tedious in case the data contains a large number of labels. Previously, this problem has been addressed using, for example, k-best beam search (Collins and Roark, 2004; Zhang and Clark, 2011; Huang et al., 2012) and parallelization (McDonald et al., 2010). In this work, we explore an alternative strategy, in which we modify the perceptron algorithm in spirit of the classic *pseudo-likelihood* approximation for ML estimation (Besag, 1975). The resulting novel algorithm has linear complexity w.r.t. the label set size and contains only a single hyper-parameter, namely, the number of passes taken over the training data set.

We evaluate the algorithm, referred to as the *pseudo-perceptron*, empirically in POS tagging on five languages. The results suggest that the approach can yield competitive accuracy compared to perceptron training accelerated using a violation-fixed 1-best beam search (Collins and Roark, 2004; Huang et al., 2012) which also provides a linear time complexity in label set size.

The rest of the paper is as follows. In Section 2, we describe the pseudo-perceptron algorithm and discuss related work. In Sections 3 and 4, we describe our experiment setup and the results, respectively. Conclusions on the work are presented in Section 5.

2 Methods

2.1 Pseudo-Perceptron Algorithm

The (unnormalized) CRF model for input and output sequences $x = (x_1, x_2, \ldots, x_{|x|})$ and

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 74–78,
Gothenburg, Sweden, April 26-30 2014.

$y = (y_1, y_2, \ldots, y_{|x|})$, respectively, is written as

$$p\left(y \mid x; \boldsymbol{w}\right) \propto \exp\Big(\boldsymbol{w} \cdot \boldsymbol{\Phi}(y, x)\Big) = \prod_{i=n}^{|x|} \exp\Big(\boldsymbol{w} \cdot \boldsymbol{\phi}(y_{i-n}, \ldots, y_i, x, i)\Big)\,, \tag{1}$$

where $\boldsymbol{w}$ denotes the model parameter vector, $\boldsymbol{\Phi}$ the vector-valued global feature extracting function, $\boldsymbol{\phi}$ the vector-valued local feature extracting function, and n the model order. We denote the tag set as $\mathcal{Y}$. The model parameters $\boldsymbol{w}$ are estimated based on training data, and test instances are decoded using the Viterbi search (Lafferty et al., 2001).

Given the model definition (1), the parameters $\boldsymbol{w}$ can be estimated in a straightforward manner using the structured perceptron algorithm (Collins, 2002). The algorithm iterates over the training set a single instance (x, y) at a time and updates the parameters according to the rule $\boldsymbol{w}^{(i)} = \boldsymbol{w}^{(i-1)} + \Delta\boldsymbol{\Phi}(x, y, z)$, where $\Delta\boldsymbol{\Phi}(x, y, z)$ for the ith iteration is written as $\Delta\boldsymbol{\Phi}(x, y, z) = \boldsymbol{\Phi}(x, y) - \boldsymbol{\Phi}(x, z)$. The prediction z is obtained as

$$z = \arg\max_{u \in \mathcal{Y}(x)} \boldsymbol{w} \cdot \boldsymbol{\Phi}(x, u) \tag{2}$$

by performing the Viterbi search over $\mathcal{Y}(x) = \mathcal{Y} \times \cdots \times \mathcal{Y}$, a product of $|x|$ copies of $\mathcal{Y}$. In case the perceptron algorithm yields a small number of incorrect predictions on the training data set, the parameters generalize well to test instances with a high probability (Collins, 2002).

The time complexity of the Viterbi search is $O(|x| \times |\mathcal{Y}|^{n+1})$. Consequently, running the perceptron algorithm can become tedious if the label set cardinality $|\mathcal{Y}|$ and/or the model order n is large. In order to speed up learning, we define a variant of the algorithm in the spirit of pseudo-likelihood (PL) learning (Besag, 1975). In analogy to PL, the key idea of the pseudo-perceptron (PP) algorithm is to obtain the required predictions over single variables y_i while fixing the remaining variables to their true values. In other words, instead of using the Viterbi search to find the z as in (2), we find a z' for each position $i \in 1..|x|$ as

$$z' = \arg\max_{u \in \mathcal{Y}'_i(x)} \boldsymbol{w} \cdot \boldsymbol{\Phi}(x, u)\,, \tag{3}$$

with $\mathcal{Y}'_i(x) = \{y_1\} \times \cdots \times \{y_{i-1}\} \times \mathcal{Y} \times \{y_{i+1}\} \times \cdots \times \{y_{|x|}\}$. Subsequent to training, test instances are decoded in a standard manner using the Viterbi search.

The appeal of PP is that the time complexity of search is reduced to $O(|x| \times |\mathcal{Y}|)$, i.e., linear in the number of labels in the label set. On the other hand, we no longer expect the obtained parameters to necessarily generalize well to test instances.[1] Consequently, we consider PP a heuristic estimation approach motivated by the rather well-established success of PL (Korč and Förstner, 2008; Sutton and McCallum, 2009).[2]

Next, we study yet another heuristic pseudo-variant of the perceptron algorithm referred to as the *piecewise-pseudo-perceptron* (PW-PP). This algorithm is analogous to the piecewise-pseudo-likelihood (PW-PL) approximation presented by Sutton and McCallum (2009). In this variant, the original graph is first split into smaller, possibly overlapping subgraphs (pieces). Subsequently, we apply the PP approximation to the pieces. We employ the approach coined *factor-as-piece* by Sutton and McCallum (2009), in which each piece contains $n + 1$ consecutive variables, where n is the CRF model order.

The PW-PP approach is motivated by the results of Sutton and McCallum (2009) who found PW-PL to increase stability w.r.t. accuracy compared to plain PL across tasks. Note that the piecewise approximation in itself is not interesting in chain-structured CRFs, as it results in same time complexity as standard estimation. Meanwhile, the PW-PP algorithm has same time complexity as PP.

2.2 Related work

Previously, impractical running times of perceptron learning have been addressed most notably using the k-best beam search method (Collins and Roark, 2004; Zhang and Clark, 2011; Huang et al., 2012). Here, we consider the "greedy" 1-best beam search variant most relevant as it shares the time complexity of the pseudo search. Therefore, in the experimental section of this work, we compare the PP and 1-best beam search.

We are aware of at least two other learning approaches inspired by PL, namely, the pseudo-max and piecewise algorithms of Sontag et al. (2010) and Alahari et al. (2010), respectively. Compared to these approaches, the PP algorithm provides a simpler estimation tool as it avoids the

[1] We leave formal treatment to future work.

[2] Meanwhile, note that pseudo-likelihood is a consistent estimator (Gidas, 1988; Hyvärinen, 2006).

hyper-parameters involved in the stochastic gradient descent algorithms as well as the regularization and margin functions inherent to the approaches of Alahari et al. (2010) and Sontag et al. (2010). On the other hand, Sontag et al. (2010) show that the pseudo-max approach achieves consistency given certain assumptions on the data generating function. Meanwhile, as discussed in previous section, we consider PP a heuristic and do not provide any generalization guarantees. To our understanding, Alahari et al. (2010) do not provide generalization guarantees for their algorithm.

3 Experimental Setup

3.1 Data

For a quick overview of the data sets, see Table 1.

Penn Treebank. The first data set we consider is the classic Penn Treebank. The complete treebank is divided into 25 sections of newswire text extracted from the Wall Street Journal. We split the data into training, development, and test sets using the sections 0-18, 19-21, and 22-24, according to the standardly applied division introduced by Collins (2002).

Multext-East. The second data we consider is the multilingual Multext-East (Erjavec, 2010) corpus. The corpus contains the novel *1984* by George Orwell. From the available seven languages, we utilize the Czech, Estonian and Romanian sections. Since the data does not have a standard division to training and test sets, we assign the 9th and 10th from each 10 consecutive sentences to the development and test sets, respectively. The remaining sentences are assigned to the training sets.

Turku Dependency Treebank. The third data we consider is the Finnish Turku Dependency Treebank (Haverinen et al., 2013). The treebank contains text from 10 different domains. We use the same data split strategy as for Multext East.

3.2 Reference Methods

We compare the PP and PW-PP algorithms with perceptron learning accelerated using 1-best beam search modified using the early update rule (Huang et al., 2012). While Huang et al. (2012) experimented with several violation-fixing methods (early, latest, maximum, hybrid), they appeared to reach termination at the same rate in POS tagging. Our preliminary experiments using the latest violation updates supported this. Consequently, we employ the early updates.

lang.	train.	dev.	test	tags	train. tags
eng	38,219	5,527	5,462	45	45
rom	5,216	652	652	405	391
est	5,183	648	647	413	408
cze	5,402	675	675	955	908
fin	5,043	630	630	2,355	2,141

Table 1: Overview on data. The training (train.), development (dev.) and test set sizes are given in sentences. The columns titled *tags* and *train. tags* correspond to total number of tags in the data set and number of tags in the training set, respectively.

We also provide results using the CRFsuite toolkit (Okazaki, 2007), which implements a 1st-order CRF model. To best of our knowledge, CRFsuite is currently the fastest freely available CRF implementation.[3] In addition to the averaged perceptron algorithm (Collins, 2002), the toolkit implements several training procedures (Nocedal, 1980; Crammer et al., 2006; Andrew and Gao, 2007; Mejer and Crammer, 2010; Shalev-Shwartz et al., 2011). We run CRFsuite using these algorithms employing their default parameters and the feature extraction scheme and stopping criterion described in Section 3.3. We then report results provided by the most accurate algorithm on each language.

3.3 Details on CRF Training and Decoding

While the methods discussed in this work are applicable for nth-order CRFs, we employ 1st-order CRFs in order to avoid overfitting the relatively small training sets.

We employ a simple feature set including word forms at position $t-2, \ldots, t+2$, suffixes of word at position t up to four letters, and three orthographic features indicating if the word at position t contains a hyphen, capital letter, or a digit.

All the perceptron variants (PP, PW-PP, 1-best beam search) initialize the model parameters with zero vectors and process the training instances in the order they appear in the corpus. At the end of each pass, we apply the CRFs using the latest averaged parameters (Collins, 2002) to the development set. We assume the algorithms have converged when the model accuracy on development

[3] See benchmark results at `http://www.chokkan.org/software/crfsuite/benchmark.html`

has not increased during last three iterations. After termination, we apply the averaged parameters yielding highest performance on the development set to test instances.

Test and development instances are decoded using a combination of Viterbi search and the *tag dictionary* approach of Ratnaparkhi (1996). In this approach, candidate tags for known word forms are limited to those observed in the training data. Meanwhile, word forms that were unseen during training consider the full label set.

3.4 Software and Hardware

The experiments are run on a standard desktop computer. We use our own C++-based implementation of the methods discussed in Section 2.

4 Results

The obtained training times and test set accuracies (measured using accuracy and out-of-vocabulary (OOV) accuracy) are presented in Table 2. The training CPU times include the time (in minutes) consumed by running the perceptron algorithm variants as well as evaluation of the development set accuracy. The column labeled *it.* corresponds to the number of passes over training set made by the algorithms before termination.

We summarize the results as follows. First, PW-PP provided higher accuracies compared to PP on Romanian, Czech, and Finnish. The differences were statistically significant[4] on Czech. Second, while yielding similar running times compared to 1-best beam search, PW-PP provided higher accuracies on all languages apart from Finnish. The differences were significant on Estonian and Czech. Third, while fastest on the Penn Treebank, the CRFsuite toolkit became substantially slower compared to PW-PP when the number of labels were increased (see Czech and Finnish). The differences in accuracies between the best performing CRFsuite algorithm and PP and PW-PP were significant on Czech.

method	it.	time (min)	acc.	OOV
English				
PP	**9**	6	96.99	87.97
PW-PP	10	7	96.98	88.11
1-best beam	17	8	96.91	88.33
Pas.-Agg.	**9**	**1**	**97.01**	**88.68**
Romanian				
PP	9	8	96.81	83.66
PW-PP	**8**	**7**	96.91	84.38
1-best beam	17	10	96.88	**85.32**
Pas.-Agg.	13	9	**97.06**	84.69
Estonian				
PP	10	8	**93.39**	78.10
PW-PP	**8**	**6**	93.35	**78.66**
1-best beam	23	15	92.95	75.65
Pas.-Agg.	15	12	93.27	77.63
Czech				
PP	**11**	26	89.37	70.67
PW-PP	16	41	89.84	72.52
1-best beam	14	**19**	88.95	70.90
Pegasos	15	341	**90.42**	**72.59**
Finnish				
PP	**11**	58	87.09	58.58
PW-PP	**11**	**56**	87.16	58.50
1-best beam	21	94	**87.38**	**59.29**
Pas.-Agg.	16	693	87.17	57.58

Table 2: Results. We report CRFsuite results provided by most accurate algorithm on each language: the *Pas.-Agg.* and *Pegasos* refer to the algorithms of Crammer et al. (2006) and Shalev-Shwartz et al. (2011), respectively.

5 Conclusions

We presented a heuristic perceptron variant for estimation of CRFs in the spirit of the classic pseudo-likelihood estimator. The resulting approximative algorithm has a linear time complexity in the label set cardinality and contains only a single hyper-parameter, namely, the number of passes taken over the training data set. We evaluated the algorithm in POS tagging on five languages. Despite its heuristic nature, the algorithm provided competetive accuracies and running times against reference methods.

Acknowledgements

This work was financially supported by Langnet (Finnish doctoral programme in language studies) and the Academy of Finland under the grant no 251170 (Finnish Centre of Excellence Program (2012-2017)). We would like to thank Dr. Onur Dikmen for the helpful discussions during the work.

[4]We establish significance (with confidence level 0.95) using the standard 1-sided Wilcoxon signed-rank test performed on 10 randomly divided, non-overlapping subsets of the complete test sets.

References

Karteek Alahari, Chris Russell, and Philip H.S. Torr. 2010. Efficient piecewise learning for conditional random fields. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 895–901.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proceedings of the 24th international conference on Machine learning*, pages 33–40.

Julian Besag. 1975. Statistical analysis of non-lattice data. *The statistician*, pages 179–195.

Michael Collins and Brian Roark. 2004. Incremental parsing with the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 111.

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing*, volume 10, pages 1–8.

Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. 2006. Online passive-aggressive algorithms. *The Journal of Machine Learning Research*, 7:551–585.

Tomaž Erjavec. 2010. Multext-east version 4: Multilingual morphosyntactic specifications, lexicons and corpora. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*.

Basilis Gidas. 1988. Consistency of maximum likelihood and pseudo-likelihood estimators for Gibbs distributions. In *Stochastic differential systems, stochastic control theory and applications*, pages 129–145.

Katri Haverinen, Jenna Nyblom, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Anna Missilä, Stina Ojala, Tapio Salakoski, and Filip Ginter. 2013. Building the essential resources for Finnish: the Turku Dependency Treebank. *Language Resources and Evaluation*.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151.

Aapo Hyvärinen. 2006. Consistency of pseudolikelihood estimation of fully visible Boltzmann machines. *Neural Computation*, 18(10):2283–2292.

Filip Korč and Wolfgang Förstner. 2008. Approximate parameter learning in conditional random fields: An empirical investigation. *Pattern Recognition*, pages 11–20.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289.

Ryan McDonald, Keith Hall, and Gideon Mann. 2010. Distributed training strategies for the structured perceptron. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 456–464.

Avihai Mejer and Koby Crammer. 2010. Confidence in structured-prediction using confidence-weighted models. In *Proceedings of the 2010 conference on empirical methods in natural language processing*, pages 971–981.

Jorge Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of conditional random fields (CRFs). *URL http://www.chokkan.org/software/crfsuite*.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the conference on empirical methods in natural language processing*, volume 1, pages 133–142.

Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. 2011. Pegasos: Primal estimated sub-gradient solver for SVM. *Mathematical Programming*, 127(1):3–30.

David Sontag, Ofer Meshi, Tommi Jaakkola, and Amir Globerson. 2010. More data means less inference: A pseudo-max approach to structured learning. In *Advances in Neural Information Processing Systems 23*, pages 2181–2189.

Charles Sutton and Andrew McCallum. 2009. Piecewise training for structured prediction. *Machine learning*, 77(2):165–194.

S.V.N. Vishwanathan, Nicol Schraudolph, Mark Schmidt, and Kevin Murphy. 2006. Accelerated training of conditional random fields with stochastic gradient methods. In *Proceedings of the 23rd international conference on Machine learning*, pages 969–976.

Yue Zhang and Stephen Clark. 2011. Syntactic processing using the generalized perceptron and beam search. *Computational Linguistics*, 37(1):105–151.

Deterministic Word Segmentation Using Maximum Matching with Fully Lexicalized Rules

Manabu Sassano
Yahoo Japan Corporation
Midtown Tower, 9-7-1 Akasaka, Minato-ku, Tokyo 107-6211, Japan
msassano@yahoo-corp.jp

Abstract

We present a fast algorithm of word segmentation that scans an input sentence in a deterministic manner just one time. The algorithm is based on simple maximum matching which includes execution of fully lexicalized transformational rules. Since the process of rule matching is incorporated into dictionary lookup, fast segmentation is achieved. We evaluated the proposed method on word segmentation of Japanese. Experimental results show that our segmenter runs considerably faster than the state-of-the-art systems and yields a practical accuracy when a more accurate segmenter or an annotated corpus is available.

1 Introduction

The aim of this study is to improve the speed of word segmentation. Applications for many Asian languages including Chinese and Japanese require word segmentation. Such languages do not have explicit word delimiters such as white spaces. Word segmentation is often needed before every task of fundamental text processing such as counting words, searching for words, indexing documents, and extracting words. Therefore, the performance of word segmentation is crucial for these languages. Take for instance, information retrieval (IR) systems for documents in Japanese. It typically uses a morphological analyzer[1] to tokenize the content of the documents. One of the most time consuming tasks in IR systems is indexing, which uses morphological analysis intensively.

Major approaches to Japanese morphological analysis (MA) are based on methods of finding the best sequence of words along with their part-of-speech tags using a dictionary where they use the Viterbi search (e.g., (Nagata, 1994), (Kudo et al., 2004)). However, computation cost of modern MA systems is mainly attributed to the Viterbi search as Kaji et al. (2010) point out.

One of methods of improving the speed of MA or word segmentation will be to avoid or reduce the Viterbi search. We can avoid this by using maximum matching in the case of word segmentation. Since there are many applications such as IR and text classification, where part-of-speech tags are not mandatory, in this paper we focus on word segmentation and adopt maximum matching for it. However, maximum matching for Japanese word segmentation is rarely used these days because the segmentation accuracy is not good enough and the accuracy of MA is much higher. In this paper we investigate to improve the accuracy of maximum-matching based word segmentation while keeping speedy processing.

2 Segmentation Algorithm

Our algorithm is basically based on maximum matching, or longest matching (Nagata, 1997). Although maximum matching is very simple and easy to implement, a segmenter with this algorithm is not sufficiently accurate. For the purpose of improving the segmentation accuracy, several methods that can be combined with maximum matching have been examined. In previous studies (Palmer, 1997; Hockenmaier and Brew, 1998), the combination of maximum matching and character-based transformational rules has been investigated for Chinese. They have reported promising results in terms of accuracy and have not mentioned the running time of their methods, which might supposedly be very slow because we have to scan an input sentence many times to apply learned transformational rules.

In order to avoid such heavy post processing, we simplify the type of rules and incorporate the process of applying rules into a single process of maximum matching for dictionary lookup. We

[1] Japanese has a conjugation system in morphology and does not put white spaces between words. Therefore, we have to do morphological analysis in order to segment a given sentence into words and give an associated part-of-speech (POS) tag to each word. In the main stream of the research of Japanese language processing, morphological analysis has meant to be a joint task of segmentation and POS tagging.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 79–83,
Gothenburg, Sweden, April 26-30 2014.

Input: c_i: sentence which is represented as a character sequence. N: the number of characters in a given sentence. t: dictionary, the data structure of which should be a trie. o_j: map an ID j to a single word or a sequence of words. h_j: total length of o_j.
Function: Lookup(t, c, i, N): search the dictionary t for the substring c starting at the position i up to N by using maximum matching. This returns the ID of the entry in the dictionary t when it matches and otherwise returns -1.
procedure Segment(c, N, t)
var i: index of the character sequence c
var j: ID of the entry in the trie dictionary t
begin
  $i \leftarrow 1$
  while ($i \leq N$) **do begin**
    $j =$ Lookup(t, c, i, N)
    if ($j = -1$) **then**
      { unknown word as a single character }
      print c_i ; $i = i + 1$
    else
      print o_j ; $i = i + h_j$
      { if o_j is a sequence of words, print each word in the sequence with a delimiter. Otherwise print o_j as a single token. }
    endif
    print delimiter
    { delimiter will be a space or something. }
  end
end

Figure 1: Algorithm of word segmentation with maximum matching that incorporates execution of transformational rules.

show in Figure 1 the pseudo code of the algorithm of word segmentation using maximum matching, where the combination of maximum matching and execution of simplified transformational rules is realized. If each of the data o_j in Figure 1 is a single token, the algorithm which is presented here is identical with segmentation by maximum matching.

We use the following types of transformational rules: $c_0c_1...c_{l-1}c_l \rightarrow w_0...w_m$ where c_i is a character and w_j is a word (or morpheme). Below are sample rules for Japanese word segmentation:

- はないか (ha-na-i-ka) → は (ha; topic-marker) ない (na-i; "does not exist") か (ka; "or") [2]
- 大工学部 (dai-ko-gaku-bu) → 大 (dai; "university") 工学部 (ko-gaku-bu; "the faculty of engineering")

Note that the form of the left hand side of the rule is the sequence of characters, not the sequence of words. Due to this simplification, we can combine dictionary lookup by maximum matching with execution of transformational rules and make them into a single process. In other words, if we find a sequence of characters of the left hand side of a certain rule, then we write out the right hand side of the rule immediately. This construction enables us to naturally incorporate execution (or application) of transformational rules into dictionary-lookup, i.e., maximum matching.

Although the algorithm in Figure 1 does not specify the algorithm or the implementation of function Lookup(), a trie is suitable for the structure of the dictionary. It is known that an efficient implementation of a trie is realized by using a double-array structure (Aoe, 1989), which enables us to look up a given key at the $O(n)$ cost, where n is the length of the key. In this case the computation cost of the algorithm of Figure 1 is $O(n)$.

We can see in Figure 1 that the Viterbi search is not executed and the average number of dictionary lookups is fewer than the number of characters of an input sentence because the average length of words is longer than one. This contrasts with Viterbi-based algorithms of word segmentation or morphological analysis that always require dictionary lookup at each character position in a sentence.

3 Learning Transformational Rules

3.1 Framework of Learning

The algorithm in Figure 1 can be combined with rules learned from a reference corpus as well as hand-crafted rules. We used here a modified version of Brill's error-driven transformation-based learning (TBL) (Brill, 1995) for rule learning.

In our system, an initial system is a word segmenter that uses maximum matching with a given

[2] If we use simple maximum matching, i.e., with no transformational rules, to segment the samples here, we will get wrong segmentations as follows: はないか → はな (ha-na; "flower") いか (i-ka; "squid"), 大工学部 → 大工 (dai-ku; "carpenter") 学部 (gaku-bu; "faculty").

$w'_0 w'_1 \cdots w'_n \rightarrow w_0 w_1 \cdots w_m$
$L\, w'_0 w'_1 \cdots w'_n \rightarrow L\, w_0 w_1 \cdots w_m$
$w'_0 w'_1 \cdots w'_n\, R \rightarrow w_0 w_1 \cdots w_m\, R$
$L\, w'_0 w'_1 \cdots w'_n\, R \rightarrow L\, w_0 w_1 \cdots w_m\, R$

Table 1: Rule templates for error-driven learning

word list and words which occur in a given reference corpus (a training corpus). Our segmenter treats an unknown word, which is not in the dictionary, as a one-character word as shown in Figure 1.

3.2 Generating Candidate Rules

In order to generate candidate rules, first we compare the output of the current system with the reference corpus and extract the differences (Tashiro et al., 1994) as rules that have the following form: $L\, w'_0 w'_1 \cdots w'_n\, R \rightarrow L\, w_0 w_1 \cdots w_m\, R$ where $w'_0 w'_1 \cdots w'_n$ is a word sequence in the system output and $w_0 w_1 \cdots w_m$ is a word sequence in the reference corpus and L is a word in the left context and R is a word in the right context. After this extraction process, we generate four lexicalized rules from each extracted rule by using the templates defined in Table 1.

3.3 Learning Rules

In order to reduce huge computation when learning a rule at each iteration of TBL, we use some heuristic strategy. The heuristic score h is defined as: $h = f * (n + m)$ where f is a frequency of the rule in question and n is the number of words in $w'_0 w'_1 \cdots w'_n$ and m is the number of words in $w_0 w_1 \cdots w_m$. After sorting the generated rules associated with the score h, we apply each candidate rule in decreasing order of h and compute the error reduction. If we get positive reduction, we obtain this rule and incorporate it into the current dictionary and then proceed to the next iteration. If we do not find any rules that reduce errors, we terminate the learning process.

4 Experiments and Discussion

4.1 Corpora and an Initial Word List

In our experiments for Japanese we used the Kyoto University Text Corpus Version 4 (we call it KC4) (Kurohashi and Nagao, 2003), which includes newspaper articles, and 470M Japanese sentences (Kawahara and Kurohashi, 2006), which is compiled from the Web. For training, we used two sets of the corpus. The first set is the articles on January 1st through 8th (7,635 sentences) of KC4. The second one is 320,000 sentences that are selected from the 470M Web corpus. Note that the Web corpus is not annotated and we use it after word segmentation is given by JUMAN 6.0 (Kurohashi and Kawahara, 2007). The test data is a set of sentences in the articles on January 9th (1,220 sentences). The articles on January 10th were used for development.

We used all the words in the dictionary of JUMAN 6.0 as an initial word list. The number of the words in the dictionary is 542,061. They are generated by removing the grammatical information such as part-of-speech tags from the entries in the original dictionary of JUMAN 6.0.

4.2 Results and Discussion

Segmentation Performance We used word based F-measure and character-wise accuracy to evaluate the segmentation performance.

Table 2 shows comparison of various systems including ours. It is natural that since our system uses only fully lexicalized rules and does not use any generalized rules, it achieves a moderate performance. However, by using the Web corpus that contains 320,000 sentences, it yields an F-measure of near 0.96, which is at the same level as the F-measure of HMMs (baseline) in (Kudo et al., 2004, Table 3). We will discuss how we can improve it in a later section.

Segmentation Speed Table 3 shows comparison of the segmentation speed of various systems for 320,000 sentences of the Web corpus. Since, in general, such comparison is heavily dependent on the implementation of the systems, we have to be careful for drawing any conclusion. However, we can see that our system, which does not use the Viterbi search, achieved considerably higher processing speed than other systems.

Further Improvement The method that we have presented so far is based on lexicalized rules. That is, we do not have any generalized rules. The system does not recognize an unknown English word as a single token because most of such words are not in the dictionary and then are split into single letters. Similarly, a number that does not appear in the training corpus is split into digits.

It is possible to improve the presented method by incorporating relatively simple post-processing that concatenates Arabic numerals, numerals in

System	# of Sent.	F-measure	Char. Acc.	# of Rules
JUMAN 6.0	NA	0.9821	0.9920	NA
MeCab 0.98 w/ jumandic	7,958	0.9861	0.9939	NA
Ours w/o training corpus	0	0.8474	0.9123	0
Ours w/ KC4	7,635	0.9470	0.9693	2228
w/ Web320K	320,000	0.9555	0.9769	24267

Table 2: Performance summary of various systems and configurations. Jumandic for MeCab (Kudo et al., 2004) is stemmed from the dictionary of JUMAN.

System (Charset Encoding)	Model/Algorithm	Time (sec.)
JUMAN 6.0 (EUC-JP)	Markov model w/ hand-tuned costs	161.09
MeCab 0.98 (UTF-8) w/ jumandic	CRFs	13.71
KyTea 0.3.3 (UTF-8) w/ jumandic	Pointwise prediction w/ SVM	188.01
Ours (UTF-8)	Maximum matching w/ rules	**3.22**

Table 3: Running time on the Web320K corpus. We used a PC (Intel Xeon 2.33 GHz with 8GB memory on FreeBSD 6.3). The model for segmentation of KyTea (Neubig et al., 2011) in our experiments is trained with the word list of JUMAN on KC4 (see in Section 4.1).

System	F-measure
JUMAN 6.0	0.9821
MeCab 0.98 w/ jumandic	**0.9861**
KyTea 0.3.3 w/ jumandic	0.9789
MEMMs (Uchimoto et al., 2001)	0.9644
HMMs (Kudo et al., 2004, Table 3)	0.9622
Ours w/ KC4	0.9470
Ours w/ KC4 + post-proc.	0.9680
Ours w/ Web320K	0.9555
Ours w/ Web320K + post-proc.	**0.9719**

Table 4: Performance comparison to other systems.

kanji[3], Latin characters, and *katakana*[4] ones. This type of post processing is commonly used in Japanese morphological analysis. JUMAN and MeCab have a similar mechanism and use it.

As an additional experiment, we incorporated this post processing into our segmenter and measured the performance. The result is shown in Table 4. The segmenter with the post processing yields an F-measure of 0.9719 when it is trained on the 320k Web corpus. We observed that the performance gap between state-of-the-art systems such as JUMAN and MeCab and ours becomes smaller. Additional computation time was +10% for the post processing and this means the segmenter with the post processing is still much faster than other sophisticated MA systems. Many applications which have to process a huge amount of documents would gain the benefits from our proposed methods.

[3] *Kanji* in Japanese, or *hanzi* in Chinese, is a ideographic script. *Kanji* means Chinese characters.

[4] *Katakana* is one of the phonetic scripts used in Japanese. It is mainly used to denote loan words and onomatopoeias. Such type of words are very productive and are often unknown words in Japanese language processing.

5 Related Work

The use of transformational rules for improving word segmentation as well as morphological analysis is not new. It is found in previous work (Papageorgiou, 1994; Palmer, 1997; Hockenmaier and Brew, 1998; Gao et al., 2004). However, their approaches require the Viterbi search and/or a heavy post process such as cascaded transformation in order to rewrite the output of the base segmenter. This leads to slow execution and systems that incorporate such approaches have much higher cost of computation than ours.

6 Conclusion

We have proposed a new combination of maximum matching and fully lexicalized transformational rules. The proposed method allows us to carry out considerably faster word segmentation with a practically reasonable accuracy. We have evaluated the effectiveness of our method on corpora in Japanese. The experimental results show that we can combine our methods with either an existing morphological analyzer or a human-edited training corpus.

References

Jun-Ichi Aoe. 1989. An efficient digital search algorithm by using a double-array structure. *IEEE Transactions on Software Engineering*, 15(9):1066–1077.

Eric Brill. 1995. Transformation-based error driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Jianfeng Gao, Andi Wu, Cheng-Ning Huang, Hong qiao Li, Xinsong Xia, and Hauwei Qin. 2004. Adaptive Chinese word segmentation. In *Proc. of ACL-2004*, pages 462–469.

Julia Hockenmaier and Chris Brew. 1998. Error-driven learning of Chinese word segmentation. In *Proc. of PACLIC 12*, pages 218–229.

Nobuhiro Kaji, Yasuhiro Fujiwara, Naoki Yoshinaga, and Masaru Kitsuregawa. 2010. Efficient staggered decoding for sequence labeling. In *Proc. of ACL 2010*, pages 485–494.

Daisuke Kawahara and Sadao Kurohashi. 2006. Case frame compilation from the web using high-performance computing. In *Proc. of LREC 2006*, pages 1344–1347.

Taku Kudo, Kaoru Yamamoto, and Yuji Matsumoto. 2004. Appliying conditional random fields to Japanese morphological analysis. In *Proc. of EMNLP 2004*, pages 230–237.

Sadao Kurohashi and Daisuke Kawahara. 2007. JUMAN (a User-Extensible Morphological Analyzer for Japanese). `http://nlp.ist.i.kyoto-u.ac.jp/index.php?JUMAN`, `http://nlp.ist.i.kyoto-u.ac.jp/EN/index.php?JUMAN`.

Sadao Kurohashi and Makoto Nagao. 2003. Building a Japanese parsed corpus. In Anne Abeille, editor, *Treebanks: Building and Using Parsed Corpora*, pages 249–260. Kluwer Academic Publishers.

Masaaki Nagata. 1994. A stochastic Japanese morphological analyzer using a forward-DP backward-A* n-best search algorithm. In *Proc. of COLING-94*, pages 201–207.

Masaaki Nagata. 1997. A self-organizing Japanese word segmenter using heuristic word identification and re-estimation. In *Proc. of WVLC-5*, pages 203–215.

Graham Neubig, Yosuke Nagata, and Shinsuke Mori. 2011. Pointwise prediction for robust, adaptable Japanese morphological analysis. In *Proc. of ACL-2011*.

David D. Palmer. 1997. A trainable rule-based algorithm for word segmentation. In *Proc. of ACL-1997*, pages 321–328.

Constantine P. Papageorgiou. 1994. Japanese word segmentation by hidden Markov model. In *Proc. of HLT-1994*, pages 283–288.

Toshihisa Tashiro, Noriyoshi Uratani, and Tsuyoshi Morimoto. 1994. Restructuring tagged corpora with morpheme adjustment rules. In *Proc. of COLING-1994*, pages 569–573.

Kiyotaka Uchimoto, Satoshi Sekine, and Hitoshi Isahara. 2001. The unknown word problem: a morphological analysis of Japanese using maximum entropy aided by a dictionary. In *Proc. of EMNLP 2001*, pages 91–99.

Painless Semi-Supervised Morphological Segmentation using Conditional Random Fields

Teemu Ruokolainen[a] **Oskar Kohonen**[b] **Sami Virpioja**[b] **Mikko Kurimo**[a]
[a] Department of Signal Processing and Acoustics, Aalto University
[b] Department of Information and Computer Science, Aalto University
firstname.lastname@aalto.fi

Abstract

We discuss data-driven morphological segmentation, in which word forms are segmented into morphs, that is the surface forms of morphemes. We extend a recent segmentation approach based on conditional random fields from purely supervised to semi-supervised learning by exploiting available unsupervised segmentation techniques. We integrate the unsupervised techniques into the conditional random field model via feature set augmentation. Experiments on three diverse languages show that this straightforward semi-supervised extension greatly improves the segmentation accuracy of the purely supervised CRFs in a computationally efficient manner.

1 Introduction

We discuss data-driven morphological segmentation, in which word forms are segmented into morphs, the surface forms of morphemes. This type of morphological analysis can be useful for alleviating language model sparsity inherent to morphologically rich languages (Hirsimäki et al., 2006; Creutz et al., 2007; Turunen and Kurimo, 2011; Luong et al., 2013). Particularly, we focus on a low-resource learning setting, in which only a small amount of annotated word forms are available for model training, while unannotated word forms are available in abundance.

We study morphological segmentation using conditional random fields (CRFs), a discriminative model for sequential tagging and segmentation (Lafferty et al., 2001). Recently, Ruokolainen et al. (2013) showed that the CRFs can yield competitive segmentation accuracy compared to more complex, previous state-of-the-art techniques. While CRFs yielded generally the highest accuracy compared to their reference methods (Poon et al., 2009; Kohonen et al., 2010), on the smallest considered annotated data sets of 100 word forms, they were outperformed by the semi-supervised Morfessor algorithm (Kohonen et al., 2010). However, Ruokolainen et al. (2013) trained the CRFs solely on the annotated data, without any use of the available unannotated data.

In this work, we extend the CRF-based approach to leverage unannotated data in a straightforward and computationally efficient manner via *feature set augmentation*, utilizing predictions of *unsupervised segmentation algorithms*. Experiments on three diverse languages show that the semi-supervised extension substantially improves the segmentation accuracy of the CRFs. The extension also provides higher accuracies on all the considered data set sizes and languages compared to the semi-supervised Morfessor (Kohonen et al., 2010).

In addition to feature set augmentation, there exists numerous approaches for semi-supervised CRF model estimation, exemplified by minimum entropy regularization (Jiao et al., 2006), generalized expectations criteria (Mann and McCallum, 2008), and posterior regularization (He et al., 2013). In this work, we employ the feature-based approach due to its simplicity and the availability of useful unsupervised segmentation methods. Varying feature set augmentation approaches have been successfully applied in several related tasks, such as Chinese word segmentation (Wang et al., 2011; Sun and Xu, 2011) and chunking (Turian et al., 2010).

The paper is organized as follows. In Section 2, we describe the CRF-based morphological segmentation approach following (Ruokolainen et al., 2013), and then show how to extend this approach to leverage unannotated data in an efficient manner. Our experimental setup and results are discussed in Sections 3 and 4, respectively. Finally,

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 84–89,
Gothenburg, Sweden, April 26-30 2014.

we present conclusions on the work in Section 5.

2 Methods

2.1 Supervised Morphological Segmentation using CRFs

We present the morphological segmentation task as a sequential labeling problem by assigning each character to one of three classes, namely {*beginning of a multi-character morph (B)*, *middle of a multi-character morph (M)*, *single character morph (S)*}. We then perform the sequential labeling using linear-chain CRFs (Lafferty et al., 2001).

Formally, the linear-chain CRF model distribution for label sequence $y = (y_1, y_2, \ldots, y_T)$ and a word form $x = (x_1, x_2, \ldots, x_T)$ is written as a conditional probability

$$p\left(y \mid x; \boldsymbol{w}\right) \propto \prod_{t=2}^{T} \exp\Big(\boldsymbol{w} \cdot \boldsymbol{\phi}(y_{t-1}, y_t, x, t)\Big)\,, \tag{1}$$

where t indexes the character positions, $\boldsymbol{w}$ denotes the model parameter vector, and $\boldsymbol{\phi}$ the vector-valued feature extracting function. The model parameters $\boldsymbol{w}$ are estimated discriminatively based on a training set of exemplar input-output pairs (x, y) using, for example, the averaged perceptron algorithm (Collins, 2002). Subsequent to estimation, the CRF model segments test word forms using the Viterbi algorithm (Lafferty et al., 2001).

We next describe the feature set $\{\phi_i(y_{t-1}, y_t, x, t)\}_{i=1}^{|\boldsymbol{\phi}|}$ by defining *emission* and *transition* features. Denoting the label set {B, M, S} as $\mathcal{Y}$, the emission feature set is defined as

$$\{\chi_m(x,t)\mathbb{1}(y_t = y'_t) \mid m \in 1..M\,, \forall y'_t \in \mathcal{Y}\}\,, \tag{2}$$

where the indicator function $\mathbb{1}(y_t = y'_t)$ returns one if and only if $y_t = y'_t$ and zero otherwise, that is

$$\mathbb{1}(y_t = y'_t) = \begin{cases} 1 & \textbf{if} \quad y_t = y'_t \\ 0 & \textbf{otherwise} \end{cases}\,, \tag{3}$$

and $\{\chi_m(x,t)\}_{m=1}^{M}$ is the set of functions describing the character position t. Following Ruokolainen et al. (2013), we employ binary functions that describe the position t of word x using all left and right substrings up to a maximum length δ. The maximum substring length δ_{max} is considered a hyper-parameter to be adjusted using a development set. While the emission features associate the input to labels, the transition feature set

$$\{\mathbb{1}(y_{t-1} = y'_{t-1})\mathbb{1}(y_t = y'_t) \mid y'_t, y'_{t-1} \in \mathcal{Y}\} \tag{4}$$

captures the dependencies between adjacent labels as irrespective of the input x.

2.2 Leveraging Unannotated Data

In order to utilize unannotated data, we explore a straightforward approach based on feature set augmentation. We exploit *predictions of unsupervised segmentation algorithms* by defining *variants of the features* described in Section 2.1. The idea is to compensate the weaknesses of the CRF model trained on the small annotated data set using the strengths of the unsupervised methods that learn from large amounts of unannotated data.

For example, consider utilizing predictions of the unsupervised Morfessor algorithm (Creutz and Lagus, 2007) in the CRF model. In order to accomplish this, we first learn the Morfessor model from the unannotated training data, and then apply the learned model on the word forms in the annotated training set. Assuming the annotated training data includes the English word *drivers*, the Morfessor algorithm might, for instance, return a (partially correct) segmentation *driv + ers*. We present this segmentation by defining a function $\upsilon(t)$, which returns 0 or 1, if the position t is in the middle of a segment or in the beginning of a segment, respectively, as in

t	1	2	3	4	5	6	7
x_t	d	r	i	v	e	r	s
$\upsilon(t)$	1	0	0	0	1	0	0

Now, given a set of U functions $\{\upsilon_u(t)\}_{u=1}^{U}$, we define variants of the emission features in (2) as

$$\{\upsilon_u(x,t)\chi_m(x,t)\mathbb{1}(y_t = y'_t) \mid \forall u \in 1..U\,, \forall m \in 1..M\,, \forall y'_t \in \mathcal{Y}\}\,. \tag{5}$$

By adding the expanded features of form (5), the CRF model learns to associate the output of the unsupervised algorithms in relation to the surrounding substring context. Similarly, an expanded transition feature is written as

$$\{\upsilon_u(x,t)\mathbb{1}(y_{t-1} = y'_{t-1})\mathbb{1}(y_t = y'_t) \mid \forall u \in 1..U\,, \forall y'_t, y'_{t-1} \in \mathcal{Y}\}\,. \tag{6}$$

After defining the augmented feature set, the CRF model parameters can be estimated in a standard manner on the small, annotated training data set. Subsequent to CRF training, the Morfessor model is applied on the test instances in order to allow the feature set augmentation and standard decoding with the estimated CRF model. We expect the Morfessor features to specifically improve

segmentation of compound words (for example, *brain+storm*), which are modeled with high accuracy by the unsupervised Morfessor algorithm (Creutz and Lagus, 2007), but can not be learned from the small number of annotated examples available for the supervised CRF training.

As another example of a means to augment the feature set, we make use of the fact that the output of the unsupervised algorithms does not have to be binary (zeros and ones). To this end, we employ the classic letter successor variety (LSV) scores presented originally by (Harris, 1955).[1] The LSV scores utilize the insight that the predictability of successive letters should be high within morph segments, and low at the boundaries. Consequently, a high variety of letters following a prefix indicates a high probability of a boundary. We use a variant of the LSV values presented by Çöltekin (2010), in which we first normalize the scores by the average score at each position t, and subseqently logarithmize the normalized value. While LSV score tracks predictability given prefixes, the same idea can be utilized for suffixes, providing the letter predecessor variety (LPV). Subsequent to augmenting the feature set using the functions $LSV(t)$ and $LPV(t)$, the CRF model learns to associate high successor and predecessor values (low predictability) to high probability of a segment boundary. Appealingly, the Harris features can be obtained in a computationally inexpensive manner, as they merely require counting statistics from the unannotated data.

The feature set augmentation approach described above is computationally efficient, if the computational overhead from the unsupervised methods is small. This is because the CRF parameter estimation is still based on the small amount of labeled examples as described in Section 2.1, while the number of features incorporated in the CRF model (equal to the number of parameters) grows linearly in the number of exploited unsupervised algorithms.

3 Experimental Setup

3.1 Data

We perform the experiments on the Morpho Challenge 2009/2010 data set (Kurimo et al., 2009; Kurimo et al., 2010) consisting of manually prepared morphological segmentations in English, Finnish and Turkish. We follow the experiment setup, including data partitions and evaluation metrics, described by Ruokolainen et al. (2013). Table 1 shows the total number of instances available for model estimation and testing.

	English	Finnish	Turkish
Train (unann.)	384,903	2,206,719	617,298
Train (ann.)	1,000	1,000	1,000
Devel.	694	835	763
Test	10,000	10,000	10,000

Table 1: Number of word types in the Morpho Challenge data set.

3.2 CRF Feature Extraction and Training

The substring features included in the CRF model are described in Section 2.1. We include all substrings which occur in the training data. The Morfessor and Harris (successor and predecessor variety) features employed by the semi-supervised extension are described in Section 2.2. We experimented on two variants of the Morfessor algorithm, namely, the Morfessor Baseline (Creutz and Lagus, 2002) and Morfessor Categories-MAP (Creutz and Lagus, 2005), CatMAP for short. The Baseline models were trained on word types and the perplexity thresholds of the CatMAP models were set equivalently to the reference runs in Morpho Challenge 2010 (English: 450, Finnish: 250, Turkish: 100); otherwise the default parameters were used. The Harris features do not require any hyper-parameters.

The CRF model (supervised and semi-supervised) is trained using the averaged perceptron algorithm (Collins, 2002). The number of passes over the training set made by the perceptron algorithm, and the maximum length of substring features are optimized on the held-out development sets.

The experiments are run on a standard desktop computer using a Python-based single-threaded CRF implementation. For Morfessor Baseline, we use the recently published implementation by Virpioja et al. (2013). For Morfessor CatMAP, we used the Perl implementation by Creutz and Lagus (2005).

[1] We also experimented on modifying the output of the Morfessor algorithm from binary to probabilistic, but these soft cues provided no consistent advantage over the standard binary output.

3.3 Reference Methods

We compare our method's performance with the fully supervised CRF model and the semi-supervised Morfessor algorithm (Kohonen et al., 2010). For semi-supervised Morfessor, we use the Python implementation by Virpioja et al. (2013).

4 Results

Segmentation accuracies for all languages are presented in Table 2. The columns titled *Train (ann.)* and *Train (unann.)* denote the number of annotated and unannotated training instances utilized by the method, respectively. To summarize, the semi-supervised CRF extension greatly improved the segmentation accuracy of the purely supervised CRFs, and also provided higher accuracies compared to the semi-supervised Morfessor algorithm[2].

Appealingly, the semi-supervised CRF extension already provided consistent improvement over the supervised CRFs, when utilizing the computationally inexpensive Harris features. Additional gains were then obtained using the Morfessor features. On all languages, highest accuracies were obtained using a combination of Harris and CatMAP features.

Running the CRF parameter estimation (including hyper-parameters) consumed typically up to a few minutes. Computing statistics for the Harris features also took up roughly a few minutes on all languages. Learning the unsupervised Morfessor algorithm consumed 3, 47, and 20 minutes for English, Finnish, and Turkish, respectively. Meanwhile, CatMAP model estimation was considerably slower, consuming roughly 10, 50, and 7 hours for English, Finnish and Turkish, respectively. Training and decoding with semi-supervised Morfessor took 21, 111, and 47 hours for English, Finnish and Turkish, respectively.

Method	Train (ann.)	Train (unann.)	F1
English			
CRF	100	0	78.8
S-MORF.	100	384,903	83.7
CRF (Harris)	100	384,903	80.9
CRF (BL+Harris)	100	384,903	82.6
CRF (CM+Harris)	100	384,903	**84.4**
CRF	1,000	0	85.9
S-MORF.	1,000	384,903	84.3
CRF (Harris)	1,000	384,903	87.6
CRF (BL+Harris)	1,000	384,903	87.9
CRF (CM+Harris)	1,000	384,903	**88.4**
Finnish			
CRF	100	0	65.5
S-MORF.	100	2,206,719	70.4
CRF (Harris)	100	2,206,719	78.9
CRF (BL+Harris)	100	2,206,719	79.3
CRF (CM+Harris)	100	2,206,719	**82.0**
CRF	1,000	0	83.8
S-MORF.	1,000	2,206,719	76.4
CRF (Harris)	1,000	2,206,719	88.3
CRF (BL+Harris)	1,000	2,206,719	88.9
CRF (CM+Harris)	1,000	2,206,719	**89.4**
Turkish			
CRF	100	0	77.7
S-MORF.	100	617,298	78.2
CRF (Harris)	100	617,298	82.6
CRF (BL+Harris)	100	617,298	84.9
CRF (CM+Harris)	100	617,298	**85.5**
CRF	1,000	0	88.6
S-MORF.	1,000	617,298	87.0
CRF (Harris)	1,000	617,298	90.1
CRF (BL+Harris)	1,000	617,298	91.7
CRF (CM+Harris)	1,000	617,298	**91.8**

Table 2: Results on test data. *CRF (BL+Harris)* denotes semi-supervised CRF extension using Morfessor Baseline and Harris features, while *CRF (CM+Harris)* denotes CRF extension employing Morfessor CatMAP and Harris features.

5 Conclusions

We extended a recent morphological segmentation approach based on CRFs from purely supervised to semi-supervised learning. We accomplished this in an efficient manner using feature set augmentation and available unsupervised segmentation techniques. Experiments on three diverse languages showed that this straightforward semi-supervised extension greatly improves the segmentation accuracy of the supervised CRFs, while being computationally efficient. The extension also outperformed the semi-supervised Morfessor algorithm on all data set sizes and languages.

[2]The improvements over the supervised CRFs and semi-supervised Morfessor were statistically significant (confidence level 0.95) according to the standard 1-sided Wilcoxon signed-rank test performed on 10 randomly divided, non-overlapping subsets of the complete test sets.

Acknowledgements

This work was financially supported by Langnet (Finnish doctoral programme in language studies) and the Academy of Finland under the Finnish Centre of Excellence Program 2012–2017 (grant no. 251170), project *Multimodally grounded language technology* (no. 254104), and LASTU Programme (nos. 256887 and 259934).

References

Michael Collins. 2002. Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, volume 10, pages 1–8. Association for Computational Linguistics.

Çagrı Çöltekin. 2010. Improving successor variety for morphological segmentation. In *Proceedings of the 20th Meeting of Computational Linguistics in the Netherlands*.

Mathias Creutz and Krista Lagus. 2002. Unsupervised discovery of morphemes. In Mike Maxwell, editor, *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, pages 21–30, Philadelphia, PA, USA, July. Association for Computational Linguistics.

Mathias Creutz and Krista Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In Timo Honkela, Ville Könönen, Matti Pöllä, and Olli Simula, editors, *Proceedings of AKRR'05, International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning*, pages 106–113, Espoo, Finland, June. Helsinki University of Technology, Laboratory of Computer and Information Science.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing*, 4(1):3:1–3:34, January.

Mathias Creutz, Teemu Hirsimäki, Mikko Kurimo, Antti Puurula, Janne Pylkkönen, Vesa Siivola, Matti Varjokallio, Ebru Arisoy, Murat Saraçlar, and Andreas Stolcke. 2007. Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *ACM Transactions on Speech and Language Processing*, 5(1):3:1–3:29, December.

Zellig Harris. 1955. From phoneme to morpheme. *Language*, 31(2):190–222.

Luheng He, Jennifer Gillenwater, and Ben Taskar. 2013. Graph-based posterior regularization for semi-supervised structured prediction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 38–46, Sofia, Bulgaria, August. Association for Computational Linguistics.

Teemu Hirsimäki, Mathias Creutz, Vesa Siivola, Mikko Kurimo, Sami Virpioja, and Janne Pylkkönen. 2006. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech and Language*, 20(4):515–541, October.

Feng Jiao, Shaojun Wang, Chi-Hoon Lee, Russell Greiner, and Dale Schuurmans. 2006. Semi-supervised conditional random fields for improved sequence segmentation and labeling. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 209–216. Association for Computational Linguistics.

Oskar Kohonen, Sami Virpioja, and Krista Lagus. 2010. Semi-supervised learning of concatenative morphology. In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages 78–86, Uppsala, Sweden, July. Association for Computational Linguistics.

Mikko Kurimo, Sami Virpioja, Ville Turunen, Graeme W. Blackwood, and William Byrne. 2009. Overview and results of Morpho Challenge 2009. In *Working Notes for the CLEF 2009 Workshop*, Corfu, Greece, September.

Mikko Kurimo, Sami Virpioja, and Ville Turunen. 2010. Overview and results of Morpho Challenge 2010. In *Proceedings of the Morpho Challenge 2010 Workshop*, pages 7–24, Espoo, Finland, September. Aalto University School of Science and Technology, Department of Information and Computer Science. Technical Report TKK-ICS-R37.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Carla E. Brodley and Andrea Pohoreckyj Danyluk, editors, *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 282–289, Williamstown, MA, USA. Morgan Kaufmann.

Minh-Thang Luong, Richard Socher, and Christopher D Manning. 2013. Better word representations with recursive neural networks for morphology. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning (CoNLL)*, pages 29–37. Association for Computational Linguistics, August.

Gideon Mann and Andrew McCallum. 2008. Generalized expectation criteria for semi-supervised learning of conditional random fields. In *Proceedings of ACL-08: HLT*, pages 870–878. Association for Computational Linguistics.

Hoifung Poon, Colin Cherry, and Kristina Toutanova. 2009. Unsupervised morphological segmentation with log-linear models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 209–217. Association for Computational Linguistics.

Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. 2013. Supervised morphological segmentation in a low-resource learning setting using conditional random fields. In *Proceedings of*

the Seventeenth Conference on Computational Natural Language Learning (CoNLL), pages 29–37. Association for Computational Linguistics, August.

Weiwei Sun and Jia Xu. 2011. Enhancing Chinese word segmentation using unlabeled data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 970–979. Association for Computational Linguistics.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.

Ville Turunen and Mikko Kurimo. 2011. Speech retrieval from unsegmented Finnish audio using statistical morpheme-like units for segmentation, recognition, and retrieval. *ACM Transactions on Speech and Language Processing*, 8(1):1:1–1:25, October.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for Morfessor Baseline. Report 25/2013 in Aalto University publication series SCIENCE + TECHNOLOGY, Department of Signal Processing and Acoustics, Aalto University.

Yiou Wang, Yoshimasa Tsuruoka Jun'ichi Kazama, Yoshimasa Tsuruoka, Wenliang Chen, Yujie Zhang, and Kentaro Torisawa. 2011. Improving Chinese word segmentation and POS tagging with semi-supervised methods using large auto-analyzed data. In *IJCNLP*, pages 309–317.

Inference of Phrase-Based Translation Models via Minimum Description Length

Jesús González-Rubio and **Francisco Casacuberta**
Departamento de Sistemas Informáticos y Computación
Universitat Politècnica de València, Camino de Vera s/n, 46021 Valencia (Spain)
{jegonzalez, fcn}@dsic.upv.es

Abstract

We present an unsupervised inference procedure for phrase-based translation models based on the minimum description length principle. In comparison to current inference techniques that rely on long pipelines of training heuristics, this procedure represents a theoretically well-founded approach to directly infer phrase lexicons. Empirical results show that the proposed inference procedure has the potential to overcome many of the problems inherent to the current inference approaches for phrase-based models.

1 Introduction

Since their introduction at the beginning of the twenty-first century, phrase-based (PB) translation models (Koehn et al., 2003) have become the state-of-the-art for statistical machine translation (SMT). PB model provide a big leap in translation quality with respect to the previous word-based translation models (Brown et al., 1990; Vogel et al., 1996). However, despite their empirical success, inference procedures for PB models rely on a long pipeline of heuristics (Och and Ney, 2003) and mismatched learning models, such as the long outperformed word-based models. Latter stages of the pipeline cannot recover mistakes or omissions made in earlier stages which forces the individual stages to massively overgenerate hypotheses. This manifests as a huge redundancy in the inferred phrase lexicons, which in turn largely penalizes the efficiency of PB systems at run-time. The fact that PB models usually cannot generate the sentence pairs in which they have been trained in, or that it is even possible to improve the performance of a PB system by discarding most of the learned phrases are clear indicators of these deficiencies (Sanchis-Trilles et al., 2011).

We introduce an unsupervised procedure to infer PB models based on the *minimum description length* (MDL) principle (Solomonoff, 1964; Rissanen, 1978). MDL, formally described in Section 2, is a general inference procedure that "learns" by "finding data regularities". MDL takes its name from the fact that regularities allow to *compress* the data, i.e. to describe it using fewer symbols than those required to describe the data literally. As such, MDL embodies a form of Occam's Razor in which the best model for a given data is the one that provides a better trade-off between goodness-of-fit on the data and "complexity" or "richness" of the model.

MDL has been previously used to infer monolingual grammars (Grünwald, 1996) and inversion transduction grammars (Saers et al., 2013). Here, we adapt the basic principles described in the latter article to the inference of PB models. The MDL inference procedure, described in Section 3, learns PB models by iteratively generalizing an initial model that perfectly overfits training data. An MDL objective is used to guide this process. MDL inference has the following desirable properties:

- Training and testing are optimized upon the same model; a basic principle of machine learning largely ignored in PB models.
- It provides a joint estimation of the structure (set of bilingual phrases) and the parameters (phrase probabilities) of PB models.
- It automatically protects against overfitting by implementing a trade-off between the expressiveness of the model and training data fitting.

The empirical evaluation described in Section 4 focuses on understanding the behavior of MDL-based PB models and their specific traits. That is, in contrast to a typical PB system building paper, we are not exclusively focused on a short term boost in translation quality. Instead, we aim at studying the adequacy and future potential of MDL as inference procedure for PB models.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 90–94,
Gothenburg, Sweden, April 26-30 2014.

2 The MDL Principle

Given a set of data $\mathcal{D}$, the MDL principle aims at obtaining the simplest possible model $\mathbf{\Phi}$ that describes $\mathcal{D}$ as well as possible (Solomonoff, 1964; Rissanen, 1978). Central to MDL is the one-to-one correspondence between description length functions and probability distributions that follows from the Kraft-McMillan inequality (McMillan, 1956). For any probability distribution $\Pr(\cdot)$, it is possible to construct a coding scheme such that the length (in bits) of the encoded data is minimum and equal to $-\log_2(\Pr(\mathcal{D}))$. In other words, searching for a minimum description length reduces to searching for a good probability distribution, and vice versa. Taking these considerations into account, MDL inference is formalized as:

$$\widehat{\mathbf{\Phi}} = \operatorname*{argmin}_{\mathbf{\Phi}} \mathrm{DL}(\mathbf{\Phi}, \mathcal{D}) \quad (1)$$

$$= \operatorname*{argmin}_{\mathbf{\Phi}} \mathrm{DL}(\mathbf{\Phi}) + \mathrm{DL}(\mathcal{D} \mid \mathbf{\Phi}) \quad (2)$$

where $\mathrm{DL}(\mathbf{\Phi})$ denotes the description length of the model, and $\mathrm{DL}(\mathcal{D} \mid \mathbf{\Phi})$ denotes the description length of the data given the model. A complete introductory tutorial of the MDL principle and methods can be found in (Grünwald, 2004).

3 MDL Phrase-Based Models

3.1 Description Length Functions

We start by defining how to compute $\mathrm{DL}(\mathbf{\Phi})$ and $\mathrm{DL}(\mathcal{D} \mid \mathbf{\Phi})$ for any PB model and data set.

Let $\Pr_{\mathbf{\Phi}}(\mathcal{D})$ be the probability of data set $\mathcal{D}$ according to PB model $\mathbf{\Phi}$. We follow the Kraft-McMillan inequality and define the description length of the data given the model as $\mathrm{DL}(\mathcal{D} \mid \mathbf{\Phi}) = -\log_2(\Pr_{\mathbf{\Phi}}(D))$, which it is the lower bound for the description length of the data.

Regarding the description length of the PB model, $\mathrm{DL}(\mathbf{\Phi})$, we compute it by serializing $\mathbf{\Phi}$ into a sequence of symbols and then computing the length of the optimal encoding of such sequence. To do that, we need one symbol for each word in the source and target languages, another symbol to separate the source and target sides in a phrase pair, and one additional symbol to distinguish between the different pairs in the phrase lexicon. For example, the following toy PB model

```
La|||The    casa|||house    azul|||blue
```

is serialized as `La|The•casa|house•azul|blue`, where symbol • separates the phrase pairs, and | separates the two sides of each pair. Assuming a uniform distribution over the K different symbols, each symbol would require $-\log_2(\frac{1}{K})$ bits to encode. We will thus require 3 bits to encode each of the 8 symbols in the example, and 33 bits to encode the whole serialized PB model (11 symbols).

3.2 Inference Procedure

We now describe how to perform the maximization in Equation (2). In the case of PB models, this reduces to a search for the optimal phrase lexicon. Obviously, an exhaustive search over all possible sets of phrase pairs in the data is unfeasible in practice. Following the ideas in (Vilar and Vidal, 2005), we implement a search procedure that iteratively generalizes an initial PB model that perfectly fits the data. Let $\mathcal{D} = \{\mathbf{f}_n, \mathbf{e}_n\}_{n=1}^{N}$ be a data set with N sentence pairs, where $\mathbf{f}_n$ are sentences in the source language and $\mathbf{e}_n$ are their corresponding translation in the target language. Our initial PB model will be as follows:

$$\mathbf{f}_1 \,|||\, \mathbf{e}_1 \quad \cdots \quad \mathbf{f}_n \,|||\, \mathbf{e}_n \quad \cdots \quad \mathbf{f}_N \,|||\, \mathbf{e}_N$$

where the probability of each pair is given by the number of occurrences of the pair in the data divided by the number of occurrences of the source (or target) language sentence.

To generalize this initial PB model, we need to identify parts of the existing phrase pairs that could be validly used in isolation. As a result, the PB model will be able to generate new translations different from the ones in the training data. From a probabilistic point of view, this process moves some of the probability mass which is concentrated in the training data out to other data still unseen; the very definition of generalization. Consider a PB model such as:

```
La casa azul|||The blue house
Esta casa azul|||This blue house
Esta casa verde|||This green house
```

It can be segmented to obtain a new PB model:

```
La|||The     casa azul|||blue house
Esta|||This  casa verde|||green house
```

which is able to generate one new sentence pair (`La casa verde→The green house`) and has a shorter description length (19 symbols) in comparison to the original model (23 symbols). We only consider segmentations that bisect the source and target phrases. More sophisticated segmentation approaches are beyond the scope of this article.

Algorithm 1 describes the proposed PB inference by iterative generalization. First, we collect the potential segmentations of the current PB

Algorithm 1: Iterative inference procedure.

input : $\mathbf{\Phi}$ (initial PB model)
output : $\widehat{\mathbf{\Phi}}$ (generalized PB model)
auxiliary : $\mathrm{collect}(\mathbf{\Phi})$ (Returns the set of possible segmentations of model $\mathbf{\Phi}$)
$\Delta\mathrm{DL}(\mathrm{s}, \mathbf{\Phi})$ (Returns variation in DL when segmenting $\mathbf{\Phi}$ according to s)
$\mathrm{sort}(\mathcal{S})$ (Sorts segmentation set $\mathcal{S}$ by variation in DL)
$\mathrm{commit}(\mathcal{S}, \mathbf{\Phi})$ (Apply segmentations in $\mathcal{S}$ to $\mathbf{\Phi}$, returns variation in DL)

```
1  begin
2      repeat
3          S ← collect(Φ);
4          candidates ← [];
5          for s ∈ S do
6              Δ' ← ΔDL(s, Φ);
7              if Δ' ≤ 0 then
8                  candidates.append({Δ', s});
9          sort(candidates);
10         Δ ← commit(candidates, Φ);
11     until Δ > 0 ;
12     return Φ;
13 end
```

model (line 3). Then, we estimate the variation in description length due to the application of each segmentation (lines 4 to 8). Finally, we sort the segmentations by variation in description length (line 9) and commit to the best of them (line 10). Specifically, given that different segmentations may modify the same phrase pair, we apply each segmentation only if it only affect phrase pairs unaffected by previous segmentations in $\mathcal{S}$. The algorithm stops when none of the segmentations lead to a reduction in description length. Saers et al., (2013) follow a similar greedy algorithm to generalize inversion transduction grammars.

The key component of Algorithm 1 is function $\Delta\mathrm{DL}(\mathrm{s}, \mathbf{\Phi})$ that evaluates the impact of a candidate segmentation s on the description length of PB model $\mathbf{\Phi}$. That is, $\Delta\mathrm{DL}(\mathrm{s}, \mathbf{\Phi})$ computes the difference in description length between the current model $\mathbf{\Phi}$ and the model $\mathbf{\Phi}'$ that would result from committing to s:

$$\begin{aligned}\Delta\mathrm{DL}(\mathrm{s}, \mathbf{\Phi}) = &\mathrm{DL}(\mathbf{\Phi}') - \mathrm{DL}(\mathbf{\Phi}) \\ &+ \mathrm{DL}(\mathcal{D} \mid \mathbf{\Phi}') - \mathrm{DL}(\mathcal{D} \mid \mathbf{\Phi}) \quad (3)\end{aligned}$$

The length difference between the phrase lexicons ($\mathrm{DL}(\mathbf{\Phi}') - \mathrm{DL}(\mathbf{\Phi})$) is trivial. We merely have to compute the difference between the lengths of the phrase pairs added and removed. The difference for the data is given by $-\log_2\left(\frac{\Pr_{\mathbf{\Phi}'}(\mathcal{D})}{\Pr_{\mathbf{\Phi}}(\mathcal{D})}\right)$, where $\Pr_{\mathbf{\Phi}'}(\mathcal{D})$ and $\Pr_{\mathbf{\Phi}}(\mathcal{D})$ are the probability of $\mathcal{D}$ according to $\mathbf{\Phi}'$ and $\mathbf{\Phi}$ respectively. These

	EuTransI (Sp / En)		
	train	tune	test
#Sentences	10k	2k	1k
#Words	97k / 99k	23k / 24k	12k / 12k
Vocabulary	687 / 513	510 / 382	571 / 435
OOV	– / –	0 / 0	0 / 0
Perplexity	– / –	8.4 / 3.4	8.1 / 3.3
	News Commentary (Sp / En)		
	train	tune	test
#Sentences	51k	2k	1k
#Words	1.4M / 1.2M	56k / 50k	30k / 26k
Vocabulary	47k / 35k	5k / 5k	8k / 7k
OOV	– / –	390 / 325	832 / 538
Perplexity	– / –	136.2 / 197.9	144.2 / 206.0

Table 1: Main figures of the experimental corpora. M and k stand for millions and thousands of elements respectively. Perplexity was calculated using 5-gram language models.

probabilities can be computed by translating the training data. However, this is a very expensive process that we cannot afford to perform for each candidate segmentation. Instead, we estimate the description length of the data in closed form based on the probabilities of the phrase pairs involved. The probability of a phrase pair $\{\tilde{\mathrm{f}}, \tilde{\mathrm{e}}\}$ is computed as the the number of occurrences of the pair divided by the number of occurrences of the source (or target) phrase. We thus estimate the probabilities in the segmented model $\mathbf{\Phi}'$ by counting the occurrences of the replaced phrase pairs as occurrences of the segmented pairs. Let $\{\tilde{\mathrm{f}}_0, \tilde{\mathrm{e}}_0\}$ be the phrase pair we are splitting into $\{\tilde{\mathrm{f}}_1, \tilde{\mathrm{e}}_1\}$ and $\{\tilde{\mathrm{f}}_2, \tilde{\mathrm{e}}_2\}$. The direct phrase probabilities in $\mathbf{\Phi}'$ will be identical to those in $\mathbf{\Phi}$ except that:

$$\begin{aligned}\mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{e}}_0 \mid \tilde{\mathrm{f}}_0) &= 0 \\ \mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{e}}_1 \mid \tilde{\mathrm{f}}_1) &= \frac{\mathrm{N}_{\mathbf{\Phi}}(\{\tilde{\mathrm{f}}_1, \tilde{\mathrm{e}}_1\}) + \mathrm{N}_{\mathbf{\Phi}}(\{\tilde{\mathrm{f}}_0, \tilde{\mathrm{e}}_0\})}{\mathrm{N}_{\mathbf{\Phi}}(\tilde{\mathrm{f}}_1) + \mathrm{N}_{\mathbf{\Phi}}(\{\tilde{\mathrm{f}}_0, \tilde{\mathrm{e}}_0\})} \\ \mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{e}}_2 \mid \tilde{\mathrm{f}}_2) &= \frac{\mathrm{N}_{\mathbf{\Phi}}(\{\tilde{\mathrm{f}}_2, \tilde{\mathrm{e}}_2\}) + \mathrm{N}_{\mathbf{\Phi}}(\{\tilde{\mathrm{f}}_0, \tilde{\mathrm{e}}_0\})}{\mathrm{N}_{\mathbf{\Phi}}(\tilde{\mathrm{f}}_2) + \mathrm{N}_{\mathbf{\Phi}}(\{\tilde{\mathrm{f}}_0, \tilde{\mathrm{e}}_0\})}\end{aligned}$$

where $\mathrm{N}_{\mathbf{\Phi}}(\cdot)$ are counts in $\mathbf{\Phi}$. Inverse probabilities are computed accordingly. Finally, we compute the variation in data description length using:

$$\begin{aligned}\frac{\Pr_{\mathbf{\Phi}'}(\mathcal{D})}{\Pr_{\mathbf{\Phi}}(\mathcal{D})} \approx &\frac{\mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{e}}_1 \mid \tilde{\mathrm{f}}_1) \cdot \mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{e}}_2 \mid \tilde{\mathrm{f}}_2)}{\mathrm{P}_{\mathbf{\Phi}}(\tilde{\mathrm{e}}_0 \mid \tilde{\mathrm{f}}_0)} \\ &\cdot \frac{\mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{f}}_1 \mid \tilde{\mathrm{e}}_1) \cdot \mathrm{P}_{\mathbf{\Phi}'}(\tilde{\mathrm{f}}_2 \mid \tilde{\mathrm{e}}_2)}{\mathrm{P}_{\mathbf{\Phi}}(\tilde{\mathrm{f}}_0 \mid \tilde{\mathrm{e}}_0)} \quad (4)\end{aligned}$$

	EUtransI		News Commentary	
	BLEU [%] (tune/test)	Size	BLEU [%] (tune/test)	Size
SotA	91.6 / 90.9	39.1k	31.4 / 30.7	2.2M
MDL	88.7 / 88.0	2.7k	24.8 / 24.6	79.1k

Table 2: Size (number of phrase pairs) of the MDL-based PB models, and quality of the generated translations. We compare against a state-of-the-art PB inference pipeline (SotA).

For a segmentation set, we first estimate the new model Φ' to reflect all the applied segmentations, and then sum the differences in description length.

4 Empirical Results

We evaluated the proposed inference procedure on the EuTransI (Amengual et al., 2000) and the News Commentary (Callison-Burch et al., 2007) corpora. Table 1 shows their main figures.

We inferred PB models (set of phrase pairs and their corresponding probabilities) with the training partitions as described in Section 3.2. Then, we included these MDL-based PB models in a conventional log-linear model optimized with the tuning partitions (Och, 2003). Finally, we generated translations for the test partitions using a conventional PB decoder (Koehn et al., 2007).

Table 2 shows size (number of phrase pairs) of the inferred MDL-based PB models, and BLEU score (Papineni et al., 2002) of their translations of the tune and test partitions. As a comparison, we display results for a state-of-the-art (SotA) PB system (Koehn et al., 2007). These results show that MDL inference obtained much more concise models (less than one tenth the number of phrases) than the standard inference pipeline. Additionally, the translations of the simple EuTransI corpus were of a similar quality as the ones obtained by the SotA system. In contrast, the quality of the translations for News Commentary was significantly lower.

To better understand these results, Figure 1 displays the histogram of phrase lengths (number of source words plus target words) of the SotA model and the MDL-based model for the News Commentaries corpus. We first observed that the length of the phrase pairs followed a completely different distribution depending on the inference procedure. Most of the phrase pairs of the MDL-based model translated one source word by one target word with an exponential decay in frequency for longer phrase pairs; a typical distribution of events in natural language (Zipf, 1935). Longer phrase pairs, about 45% of the total, contain sequences of words that only appear once in the corpus, and thus, they cannot be segmented in any way that leads to a reduction in description length. Although formally correct, long phrase pairs generalize poorly which explains the comparatively poor performance of MDL inference for the News Commentaries corpus. This problem was largely attenuated for EuTransI due to its simplicity.

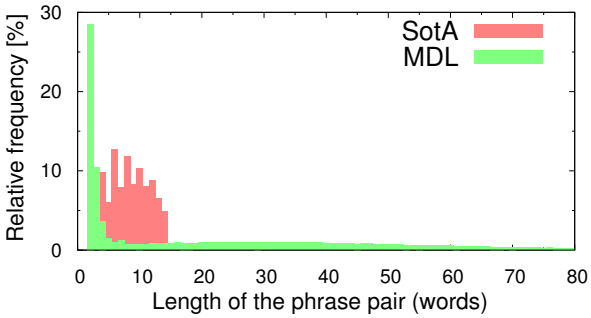


Figure 1: Histogram of lengths (source plus target words) for the phrase pairs in the inferred models.

5 Conclusions and Future Developments

We have described a simple, unsupervised inference procedure for PB models that learns phrase lexicons by iteratively splitting existing phrases into smaller phrase pairs using a theoretically well-founded minimum description length objective. Empirical results have shown that the inferred PB models, far from the artificial redundancy of the conventional PB inference pipeline, are very parsimonious and provide competitive translations for simple translation tasks.

The proposed methodology provides a solid foundation from where to develop new PB inference approaches that overcome the problems inherent to the long pipeline of heuristics that nowadays constitute the state-of-the-art. Future developments in this direction will include:

- A more sophisticated segmentation procedure that allow to divide the phrases into more that two segments.
- A hybrid approach where the long phrase pairs remaining after the MDL inference are further segmented, e.g., according to a word lexicon.
- The inclusion of lexical models in the definition of the PB model.

Acknowledgments

Work supported by the European Union 7^{th} Framework Program (FP7/2007-2013) under the CasMaCat project (grans agreement n$^{\text{o}}$ 287576), by Spanish MICINN under grant TIN2012-31723, and by the Generalitat Valenciana under grant ALMPR (Prometeo/2009/014).

References

Juan-Carlos Amengual, M. Asunción Castaño, Antonio Castellanos, Víctor M. Jiménez, David Llorens, Andrés Marzal, Federico Prat, Juan Miguel Vilar, José-Miguel Benedí, Francisco Casacuberta, Moisés Pastor, and Enrique Vidal. 2000. The eutrans spoken language translation system. *Machine Translation*, 15(1-2):75–103.

Peter F. Brown, John Cocke, Stephen A. Della Pietra, Vincent J. Della Pietra, Fredrick Jelinek, John D. Lafferty, Robert L. Mercer, and Paul S. Roossin. 1990. A statistical approach to machine translation. *Computational Linguistics*, 16:79–85.

Chris Callison-Burch, Cameron Fordyce, Philipp Koehn, Christof Monz, and Josh Schroeder. 2007. (Meta-) evaluation of machine translation. In *Proceedings of the Workshop on Statistical Machine Translation*, pages 136–158.

Peter Grünwald. 1996. A minimum description length approach to grammar inference. *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*, pages 203–216.

Peter Grünwald. 2004. A tutorial introduction to the minimum description length principle. http://arxiv.org/abs/math/0406077.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 48–54.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the Association for Computational Linguistics, demonstration session*, June.

Brockway McMillan. 1956. Two inequalities implied by unique decipherability. *IRE Transactions on Information Theory*, 2(4):115–116.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the Meeting on Association for Computational Linguistics*, pages 160–167. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the Meeting on Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics.

Jorma Rissanen. 1978. Modeling by shortest data description. *Automatica*, 14(5):465 – 471.

Markus Saers, Karteek Addanki, and Dekai Wu. 2013. Iterative rule segmentation under minimum description length for unsupervised transduction grammar induction. In *Statistical Language and Speech Processing*, volume 7978 of *Lecture Notes in Computer Science*, pages 224–235. Springer.

Germán Sanchis-Trilles, Daniel Ortiz-Martínez, Jesús González-Rubio, Jorge González, and Francisco Casacuberta. 2011. Bilingual segmentation for phrasetable pruning in statistical machine translation. In *Proceedings of the 15th Conference of the European Association for Machine Translation*.

Ray Solomonoff. 1964. A formal theory of inductive inference, parts 1 and 2. *Information and Control*, 7:1–22, 224–254.

Juan Miguel Vilar and Enrique Vidal. 2005. A recursive statistical translation model. In *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pages 199–207.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th conference on Computational linguistics*, pages 836–841.

George Kingsley Zipf. 1935. *The Psychobiology of Language*. Houghton-Mifflin.

Chinese Native Language Identification

Shervin Malmasi
Centre for Language Technology
Macquarie University
Sydney, NSW, Australia
shervin.malmasi@mq.edu.au

Mark Dras
Centre for Language Technology
Macquarie University
Sydney, NSW, Australia
mark.dras@mq.edu.au

Abstract

We present the first application of Native Language Identification (NLI) to non-English data. Motivated by theories of language transfer, NLI is the task of identifying a writer's native language (L1) based on their writings in a second language (the L2). An NLI system was applied to Chinese learner texts using topic-independent syntactic models to assess their accuracy. We find that models using part-of-speech tags, context-free grammar production rules and function words are highly effective, achieving a maximum accuracy of 71% . Interestingly, we also find that when applied to equivalent English data, the model performance is almost identical. This finding suggests a systematic pattern of cross-linguistic transfer may exist, where the degree of transfer is independent of the L1 and L2.

1 Introduction

Native Language Identification (NLI) is the task of identifying an author's native language (L1) based on their writings in a second language (the L2). NLI works by identifying language use patterns that are common to groups of speakers that share the same native language. This process is underpinned by the presupposition that an author's L1 will dispose them towards particular language production patterns in their L2, as influenced by their mother tongue. This relates to Cross-Linguistic Influence (CLI), a key topic in the field of Second Language Acquisition (SLA) that analyzes transfer effects from the L1 on later learned languages (Ortega, 2009).

While NLI has applications in security, most research has a strong linguistic motivation relating to language teaching and learning. Rising numbers of language learners have led to an increasing need for language learning resources, which has in turn fuelled much of the language acquisition research of the past decade. In this context, by identifying L1-specific language usage and error patterns, NLI can be used to better understand SLA and develop teaching methods, instructions and learner feedback that is specific to their mother tongue.

However, all of the NLI research to date has focused exclusively on English L2 data. To this end there is a need to apply NLI to other languages, not only to gauge their applicability but also to aid in teaching research for other emerging languages.

Interest in learning Chinese is rapidly growing, leading to increased research in Teaching Chinese as a Second Language (TCSL) and the development of related resources such as learner corpora (Chen et al., 2010). The application of these tools and scientific methods like NLI can greatly assist researchers in creating effective teaching practices and is an area of active research.

The aim of this research is to evaluate the cross-language applicability of NLI techniques by applying them to Chinese learner texts, evaluating their efficacy and comparing the results with their English equivalents.

To the best of our knowledge this is the first reported application of NLI to non-English data and we believe this is an important step in gaining deeper insights about the technique.

2 Related Work

NLI is a fairly recent, but rapidly growing area of research. While some research was conducted in the early 2000s, the most significant work has only appeared in the last few years (Wong and Dras, 2009; Wong and Dras, 2011; Swanson and Charniak, 2012; Tetreault et al., 2012; Bykh and Meurers, 2012).

Most studies approach NLI as a multi-class supervised classification task. In this experimental design, the L1 metadata are used as class labels

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 95–99,
Gothenburg, Sweden, April 26-30 2014.

and the individual writings are used as training and testing data. Using lexical and syntactic features of increasing sophistication, researchers have obtained good results under this paradigm. While a detailed exposition of NLI has been omitted here due to space constraints, a concise review can be found in Bykh and Meurers (2012).

2.1 NLI 2013 Shared Task

This increased interest brought unprecedented level of research focus and momentum, resulting in the first NLI shared task being held in 2013.[1] The shared task aimed to facilitate the comparison of results by providing a large NLI-specific dataset and evaluation procedure, to enable direct comparison of results achieved through different methods. Overall, the event was considered a success, drawing 29 entrants and experts from not only Computational Linguistics, but also SLA. The best teams achieved accuracies of greater than 80% on this 11-class classification task. A detailed summary of the results is presented in Tetreault et al. (2013).

3 Data

Growing interest has led to the recent development of the Chinese Learner Corpus (Wang et al., 2012), the first large-scale corpus of learner texts comprised of essays written by university students. Learners from 59 countries are represented and proficiency levels have been sampled representatively across beginners, intermediate and advanced learners. However, texts by native speakers of other Asian countries are disproportionately represented, likely due to geographical proximity.

For this work we extracted 3.75 million tokens of text from the CLC in the form of individual sentences.[2] Following the methodology of Brooke and Hirst (2011), we combine the sentences from the same L1 to form texts of 600 tokens on average, creating a set of documents suitable for NLI[3].

We choose the top 11 languages, shown in Table 1, to use in our experiments. This is due to two considerations. First, while many L1s are represented in the corpus, most have relatively few texts. Choosing the top 11 classes allows us to balance having a large number of classes, and also maximizes the amount of data used. Secondly, this is the same number of classes used in the NLI 2013 shared task, enabling us to draw cross-language comparisons with the shared task results.

Language	Size	Language	Size
Filipino FIL	415	Indonesian IND	402
Thai THA	400	Laotian LAO	366
Burmese MYA	349	Korean* KOR	330
Khmer KHM	294	Vietnamese VIE	267
Japanese* JAP	180	Spanish* SPA	112
Mongolian MON	101		

Table 1: Our data, broken down by language and the number of texts in each class. Languages overlapping with the TOEFL11 corpus marked with *.

4 Experimental Setup

We also follow the supervised classification approach described in §2. We devise and run experiments using several models that capture different types of linguistic information. For each model, features are extracted from the texts and a classifier is trained to predict the L1 labels using the features. As our data is not topic-balanced, we avoid using topic-dependent lexical features such as character or word n-grams.

Each experiment is run with two feature representations: binary (presence/absence of a feature) and normalized frequencies, where feature values are normalized to text length using the $l2$-norm.

4.1 Parser

The Stanford CoreNLP[4] suite of NLP tools and the provided Chinese models are used to tokenize, PoS tag and parse the unsegmented corpus texts.

4.2 Classifier

We use Support Vector Machines for classification. Specifically, we use the LIBLINEAR SVM package (Fan et al., 2008) as it is well-suited to text classification tasks with large numbers of features and texts. We use the L2-regularized L2-loss support vector classification (dual) solver.

4.3 Evaluation

The same evaluation metrics and standards used in the NLI2013 Shared Task are used: we report classification accuracy under 10-fold cross-validation. We also use the same number of classes as the shared task to facilitate comparative analyses.

[1] Organised by the Educational Testing Service and co-located with the eighth instalment of the Building Educational Applications Workshop at NAACL/HLT 2013. sites.google.com/site/nlisharedtask2013/

[2] Full texts are not made available, only individual sentences with the relevant metadata (proficiency/nationality).

[3] Pending permission from the CLC corpus authors, we will attempt to release the Chinese NLI dataset publicly.

[4] http://nlp.stanford.edu/software/corenlp.shtml

Feature	Accuracy (%)	
	Binary	Frequency
Random Baseline	9.09	9.09
PoS unigrams	20.12	35.32
Part-of-Speech bigrams	32.83	54.24
Part-of-Speech trigrams	47.24	55.60
Function Words	43.93	51.91
Production Rules	36.14	49.80
All features	61.75	70.61

Table 2: Chinese Native Language Identification accuracy (%) for all of our models.

5 Experiments and Results

5.1 Part-of-Speech tag n-grams

Our first experiment assesses the utility of the syntactic information captured by part-of-speech (PoS) tags for Chinese NLI. The PoS tags for each text are predicted and n-grams of size 1–3 are extracted from the tags. These n-grams capture (very local) syntactic patterns of language use and are used as classification features.

The results for these three features, and our other models are shown in Table 2. The trigram frequencies give the best accuracy of 55.60%, suggesting that there exist group-specific patterns of Chinese word order and category choice which provide a highly discriminative cue about the L1.

5.2 Function Words

As opposed to content words, function words are topic-independent grammatical words that indicate the relations between other words. They include determiners, conjunctions and auxiliary verbs. Distributions of English function words have been found to be useful in studies of authorship attribution and NLI. Unlike PoS tags, this model analyzes the author's specific word choices.

We compiled a list of 449 Chinese function words[5] to be used as features in this model. As shown in Table 2, the function word frequency features provide the best accuracy of 51.91%, significantly higher than the random baseline. This again suggests the presence of L1-specific grammatical and lexical choice patterns that can help distinguish the L1, potentially due to cross-linguistic transfer. Such lexical transfer effects have been previously noted by researchers and linguists (Odlin, 1989). These effects are mediated not only by cognates and similarities in word forms, but also word semantics and meanings.

[5]The function word list was compiled from Chinese language teaching resources. The complete list can be accessed at `http://comp.mq.edu.au/~madras/research/data/chinese-fw.txt`

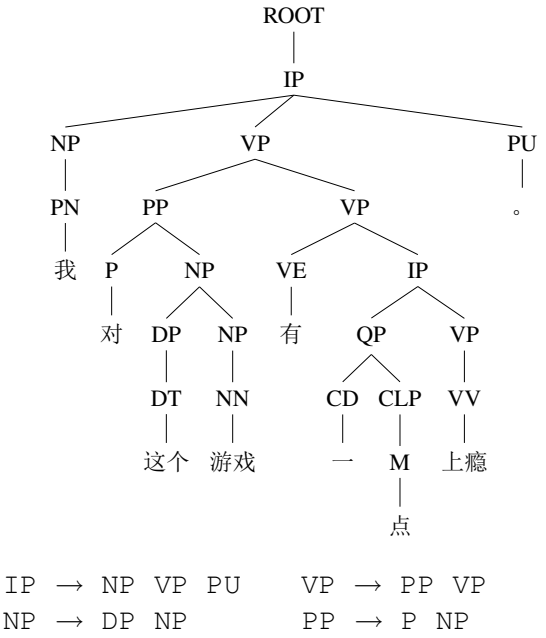


Figure 1: A constituent parse tree for a sentence from the corpus along with some of the context-free grammar production rules extracted from it.

5.3 Context-free Grammar Production Rules

In the next experiment we investigate the differences in the distribution of the context-free grammar production rules used by the learners. To do this, constituent parses for all sentences are obtained and the production rules, excluding lexicalizations, are extracted. Figure 1 shows a sample tree and rules. These context-free phrase structure rules capture the overall structure of grammatical constructions and are used as classification features in this experiment.

As seen in Table 2, the model achieves an accuracy of 49.80%. This supports the hypothesis that the syntactic substructures contain characteristic constructions specific to L1 groups and that these syntactic cues strongly signal the writer's L1.

5.4 Combining All Features

Finally, we assess the redundancy of the information captured by our models by combining them all into one vector space to create a single classifier. From Table 2 we see that for each feature representation, the combined feature results are higher than the single best feature, with a max-

imum accuracy of 70.61%. This demonstrates that for at least some of the features, the information they capture is orthogonal and complementary, and combining them can improve results.

6 Discussion

A key finding here is that NLI models can be successfully applied to non-English data. This is an important step for furthering NLI research as the field is still relatively young and many fundamental questions have yet to be answered.

All of the tested models are effective, and they appear to be complementary as combining them improves overall accuracy. We also note the difference in the efficacy of the feature representations and see a clear preference for frequency-based feature values. Others have found that binary features are the most effective for English NLI (Brooke and Hirst, 2012), but our results indicate frequency information is more informative in this task. The combination of both feature types has also been reported to be effective (Malmasi et al., 2013).

To see how these models perform across languages, we also compare the results against the TOEFL11 corpus used in the NLI2013 shared task. We perform the same experiments on that dataset using the English CoreNLP models, Penn Treebank PoS tagset and a set of 400 English function words. Figure 2 shows the results side by side.

Remarkably, we see that the model results closely mirror each other across corpora. This is a highly interesting finding from our study that merits further investigation. There is a systematic pattern occurring across data from learners of completely different L1-L2 pairs. This suggests that manifestations of CLI via surface phenomena occur at the same levels and patternings regardless of the L2. Cross-language studies can help researchers in linguistics and cognitive science to better understand the SLA process and language transfer effects. They can enhance our understanding of how language is processed in the brain in ways that are not possible by just studying monolinguals or single L1-L2 pairs, thereby providing us with important insights that increase our knowledge and understanding of the human language faculty.

One limitation of this work is the lack of similar amounts of training data for each language. However, many of the early and influential NLI studies (e.g. Koppel et al. (2005), Tsur and Rappoport (2007)) were performed under similar circumstances. This issue was noted at the time, but did not deter researchers as corpora with similar issues were used for many years. Non-English NLI is also at a similar state where the extant corpora are not optimal for the task, but no other alternatives exist for conducting this research.

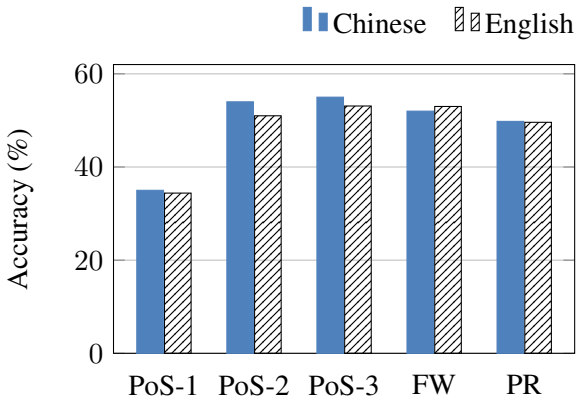


Figure 2: Comparing feature performance on the Chinese Learner Corpus and English TOEFL11 corpora. PoS-1/2/3: PoS uni/bi/trigrams, FW: Function Words, PR: Production Rules

Finally, there are also a number of way to further develop this work. Firstly, the experimental scope could be expanded to use even more linguistically sophisticated features such as dependency parses. Model accuracy could potentially be improved by using the metadata to develop proficiency-segregated models. Classifier ensembles could also help in increasing the accuracy.

7 Conclusion

In this work we have presented the first application of NLI to non-English data. Using the Chinese Learner Corpus, we compare models based on PoS tags, function words and context-free grammar production rules and find that they all yield high classification accuracies.

Comparing the models against an equivalent English learner corpus we find that the accuracies are almost identical across both L2s, suggesting a systematic pattern of cross-linguistic transfer where the degree of transfer is independent of the L1 and L2. Further research with other L2 learner corpora is needed to investigate this phenomena.

Acknowledgments

We wish to thank Associate Professor Maolin Wang for providing access to the CLC corpus, and Zhendong Zhao for his assistance. We also thank the reviewers for their constructive feedback.

References

Julian Brooke and Graeme Hirst. 2011. Native language detection with 'cheap' learner corpora. In *Conference of Learner Corpus Research (LCR2011)*, Louvain-la-Neuve, Belgium. Presses universitaires de Louvain.

Julian Brooke and Graeme Hirst. 2012. Robust, Lexicalized Native Language Identification. In *Proceedings of COLING 2012*, pages 391–408, Mumbai, India, December. The COLING 2012 Organizing Committee.

Serhiy Bykh and Detmar Meurers. 2012. Native Language Identification using Recurring n-grams – Investigating Abstraction and Domain Dependence. In *Proceedings of COLING 2012*, pages 425–440, Mumbai, India, December. The COLING 2012 Organizing Committee.

Jianguo Chen, Chuang Wang, and Jinfa Cai. 2010. *Teaching and learning Chinese: Issues and perspectives*. IAP.

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

Moshe Koppel, Jonathan Schler, and Kfir Zigdon. 2005. Automatically determining an anonymous author's native language. In *Intelligence and Security Informatics*, volume 3495 of *LNCS*, pages 209–217. Springer-Verlag.

Shervin Malmasi, Sze-Meng Jojo Wong, and Mark Dras. 2013. Nli shared task 2013: Mq submission. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 124–133, Atlanta, Georgia, June. Association for Computational Linguistics.

Terence Odlin. 1989. *Language Transfer: Cross-linguistic Influence in Language Learning*. Cambridge University Press, Cambridge, UK.

Lourdes Ortega. 2009. *Understanding Second Language Acquisition*. Hodder Education, Oxford, UK.

Benjamin Swanson and Eugene Charniak. 2012. Native Language Detection with Tree Substitution Grammars. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 193–197, Jeju Island, Korea, July. Association for Computational Linguistics.

Joel Tetreault, Daniel Blanchard, Aoife Cahill, and Martin Chodorow. 2012. Native tongues, lost and found: Resources and empirical evaluations in native language identification. In *Proceedings of COLING 2012*, pages 2585–2602, Mumbai, India, December. The COLING 2012 Organizing Committee.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 48–57, Atlanta, Georgia, June. Association for Computational Linguistics.

Oren Tsur and Ari Rappoport. 2007. Using classifier features for studying the effect of native language on the choice of written second language words. In *Proc. Workshop on Cognitive Aspects of Computat. Language Acquisition*, pages 9–16.

Maolin Wang, Qi Gong, Jie Kuang, and Ziyu Xiong. 2012. The development of a chinese learner corpus. In *Speech Database and Assessments (Oriental COCOSDA), 2012 International Conference on*, pages 1–6. IEEE.

Sze-Meng Jojo Wong and Mark Dras. 2009. Contrastive Analysis and Native Language Identification. In *Proceedings of the Australasian Language Technology Association Workshop 2009*, pages 53–61, Sydney, Australia, December.

Sze-Meng Jojo Wong and Mark Dras. 2011. Exploiting Parse Structures for Native Language Identification. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1600–1610, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Unsupervised Parsing for Generating Surface-Based Relation Extraction Patterns

Jens Illig
University of Kassel
Wilhelmshöher Allee 73
D-34121 Kassel, Germany
illig@cs.uni-kassel.de

Benjamin Roth and **Dietrich Klakow**
Saarland University
D-66123 Saarbrücken, Germany
{benjamin.roth, dietrich.klakow}
@lsv.uni-saarland.de

Abstract

Finding the right features and patterns for identifying relations in natural language is one of the most pressing research questions for relation extraction. In this paper, we compare patterns based on supervised and unsupervised syntactic parsing and present a simple method for extracting surface patterns from a parsed training set. Results show that the use of surface-based patterns not only increases extraction speed, but also improves the quality of the extracted relations. We find that, in this setting, unsupervised parsing, besides requiring less resources, compares favorably in terms of extraction quality.

1 Introduction

Relation extraction is the task of automatically detecting occurrences of expressed relations between entities in a text and structuring the detected information in a tabularized form. In natural language, there are infinitely many ways to creatively express a set of semantic relations in accordance to the syntax of the language. Languages vary across domains and change over time. It is therefore impossible to statically capture all ways of expressing a relation.

Most relation extraction systems (Bunescu and Mooney, 2005; Snow et al., 2005; Zhang et al., 2006; Mintz et al., 2009; Alfonseca et al., 2012; Min et al., 2012) generalize semantic relations by taking into account statistics about the syntactic construction of sentences. Usually supervised parsers are applied for parsing sentences.

Statistics are then utilized to *machine-learn* how textual mentions of relations can be identified. Many researchers avoid the need for expensive corpora with manually labeled relations by applying a scheme called *distant supervision* (Mintz et al., 2009; Roth et al., 2013) which hypothesizes that all text fragments containing argument co-occurrences of known semantic relation facts indeed express these relations. Still, systems relying on supervised parsers require training from annotated treebanks, which are expensive to create, and highly domain- and language dependent when available.

An alternative is unsupervised parsing, which automatically induces grammars by structurally analyzing unlabeled corpora. Applying unsupervised parsing thus avoids the limitation to languages and domains for which annotated data is available. However, induced grammars do not match traditional linguistic grammars. In most of the research on parsing, unsupervised parsers are still evaluated based on their level of correspondence to treebanks. This is known to be problematic because there are several different ways of linguistically analyzing text, and treebank annotations also contain questionable analyses (Klein, 2005). Moreover, it is not guaranteed that the syntactic analysis which is most conforming to a general linguistic theory is also best suited in an extrinsic evaluation, such as for relation extraction.

In this work, we apply a supervised and an unsupervised parser to the relation extraction task by extracting statistically counted patterns from the resulting parses. By utilizing the performance of the overall relation extraction system as an indirect measure of a parser's practical qualities, we get a task-driven evaluation comparing supervised and unsupervised parsers. To the best of our knowledge, this is the first work to compare general-purpose unsupervised and supervised parsing on the application of relation extraction. Moreover, we introduce a simple method to obtain shallow patterns from syntactic analyses and show that, besides eliminating the need to parse text during system application, such patterns also increase extraction quality. We discover that, for this method, un-

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 100–105,
Gothenburg, Sweden, April 26-30 2014.

supervised parsing achieves better extraction quality than the more expensive supervised parsing.

1.1 Related Work

Unsupervised and weakly supervised training methods have been applied to relation extraction (Mintz et al., 2009; Banko et al., 2007; Yates and Etzioni, 2009) and similar applications such as semantic parsing (Poon and Domingos, 2009) and paraphrase acquisition (Lin and Pantel, 2001). However, in such systems, parsing is commonly applied as a separately trained subtask[1] for which supervision is used.

Hänig and Schierle (2009) have applied unsupervised parsing to a relation extraction task but their task-specific data prohibits supervised parsing for comparison.

Unsupervised parsing is traditionally only evaluated intrinsically by comparison to gold-standard parses. In contrast, Reichart and Rappoport (2009) count POS token sequences inside sub-phrases for measuring parsing consistency. But this count is not clearly related to application qualities.

2 Methodology

A complete relation extraction system consists of multiple components. Our system follows the architecture described by Roth et al. (2012). In short, the system retrieves queries in the form of entity names for which all relations captured by the system are to be returned. The entity names are expanded by alias-names extracted from Wikipedia link anchor texts. An information retrieval component retrieves documents containing either the name or one of the aliases. Further filtering retains only sentences where a named entity tagger labeled an occurrence of the queried entity as being of a suitable type and furthermore found a possible entity for the relation's second argument. For each candidate sentence, a classifier component then identifies whether one of the captured relation types is expressed and, if so, which one it is. Postprocessing then outputs the classified relation according to task-specific format requirements. Here, we focus on the relation type classifier.

[1] An exception is the joint syntactic and semantic (supervised) parsing model inference by Henderson et al. (2013)

2.1 Pattern Extraction

For our relation extraction system, we use a simple pattern matching framework. Whenever at least one candidate sentence containing two entities A and B matches one of the patterns extracted for a certain relation type R, the classifier states that R holds between A and B.

We experimented with two types of patterns. First, we simply parsed the training set and extracted shortest dependency path patterns. These patterns search for matches on the parse tree. Following Lin and Pantel (2001), the shortest path connecting two arguments in a dependency graph has been widely used as a representation of relation instance mentions. The general idea is that shortest paths skip over irrelevant optional parts of a sentence such as in *$1, who ... founded $2* where the shortest path pattern `$1←founded→$2` matches although an irrelevant relative clause appears between the arguments $1 and $2. Similar representations have been used by Mintz et al. (2009), Alfonseca et al. (2012) and Snow et al. (2005).

In a second set of experiments, we used the shortest dependency paths in parsed training sentences to generate surface-based patterns. These patterns search for matches directly on plain text and therefore do no longer rely on parsing at application time. The patterns are obtained by turning the shortest paths between relational arguments in the parsed training data into token sequences with gaps. The token sequences consist of all words in the sentence that appear on the shortest dependency path. Argument positions in the surface patterns are specified by special tokens `$1` and `$2`. At all places, where there are one or more tokens which are not on the shortest dependency path but which are surrounded either by tokens on the dependency path or by arguments, an asterisk represents up to four unspecified tokens. For the shortest path `$1←,←who→$2` connecting *Friedman* and *economist* in the DMV parse depicted in Figure 1, this method generates the pattern `$1, * $2 who`. As can be seen, such patterns can capture a conjunction of token presence conditions to the left, between, and to the right of the arguments. In cases where argument entities are not parsed as a single complete phrase, we generate patterns for each possible combination of outgoing edges from the two arguments. We dismiss patterns generated for less than four distinct argument entity pairs of

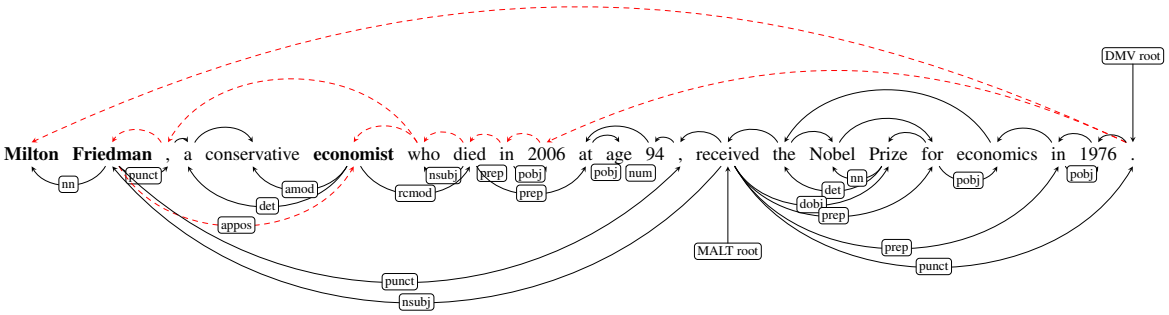


Figure 1: Comparison of a DMV (above text) and a MALT parse (below text) of the same sentence.

the same relation type. For each pattern, we calculate the precision on the training set and retain only patterns above a certain precision threshold.

2.2 Supervised and Unsupervised Parsing

Typical applications which require syntactic analyses make use of a parser that has been trained under supervision of a labeled corpus conforming to a linguistically engineered grammar. In contrast, unsupervised parsing induces a grammar from frequency structures in plain text.

Various algorithms for unsupervised parsing have been developed in the past decades. Headden (2012) gives a rather recent and extensive overview of unsupervised parsing models. For our work, we use the Dependency Model with Valence (DMV) by Klein and Manning (2004). Most of the more recent unsupervised dependency parsing research is based on this model. DMV is a generative head-outward parsing model which is trained by expectation maximization on part-of-speech (POS) sequences of the input sentences. Starting from a single root token, head tokens generate dependants by a probability conditioned on the direction (left/right) from the head and the head's token type. Each head node generates tokens until a stop event is generated with a probability dependent on the same criteria plus a flag whether some dependant token has already been generated in the same direction.

For comparison of unsupervised and supervised parsing, we apply the (Nivre, 2003) deterministic incremental parsing algorithm *Nivre arc-eager*, the default algorithm of the *MALT* framework[2] (Nivre et al., 2007). In this model, for each word token, an SVM classifier decides for a parser state transition, which, in conjunction with other decisions, determines where phrases begin and end.

[2] `http://www.maltparser.org` as of Nov. 2013

3 Experiments

We used the plain text documents of the English Newswire and Web Text Documents provided for TAC KBP challenge 2011 (Ji et al., 2011). We automatically annotated relation type mentions in these documents by distant supervision using the online database Freebase[3], i.e. for all relation types of TAC KBP 2011, we took relation triples from Freebase and, applying preprocessing as described in Section 2, we retrieved sentences mentioning both arguments of some Freebase relation with matching predicted entity types. We hypothesize that all sentences express the respective Freebase relation. This way we retrieved a distantly supervised training set of 480 622 English sentences containing 92 468 distinct relation instances instantiating 41 TAC KBP relation types.

3.1 Training and Evaluation

From our retrieved set of sentences, we took those with a maximum length of 10 tokens and transformed them to POS sequences. We trained DMV only on this dataset of short POS sequences, which we expect to form mentions of a modeled relation. Therefore, we suspect that DMV training assigns an increased amount of probability mass to dependency paths along structures which are truly related to these relations. We used the DMV implementation from Cohen and Smith (2009) [4].

For the supervised *Nivre arc-eager* parser we used *MALT* (Nivre et al., 2007) with a pre-trained Penn Treebank (Marcus et al., 1993) model[5]. As a baseline, we tested *left branching* parses i.e.

[3] `http://www.freebase.com` as of Nov. 2013

[4] publicly available at `http://www.ark.cs.cmu.edu/DAGEEM/` as of Nov. 2013 (parser version 1.0).

[5] `http://www.maltparser.org/mco/english_parser/engmalt.linear-1.7.mco` as of Nov. 2013

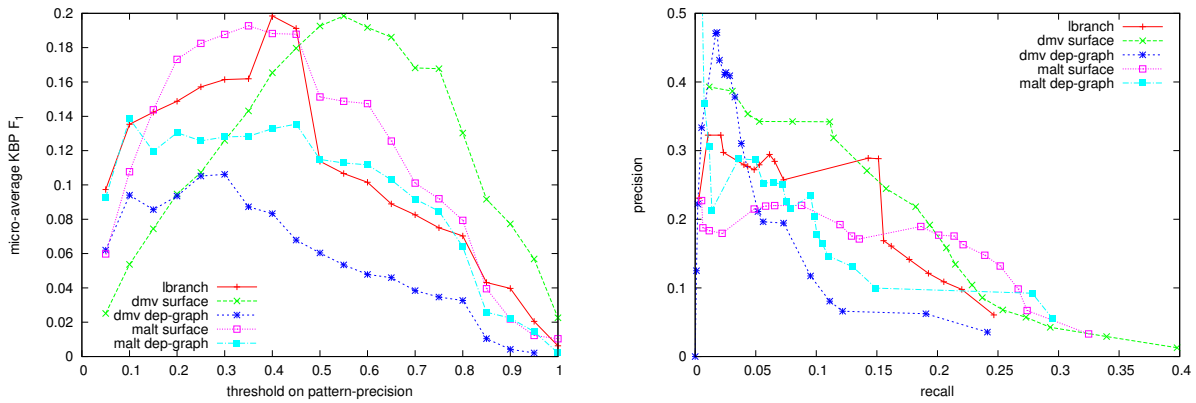


Figure 2: micro-averaged F_1 and precision&recall results for varied training precision thresholds

pattern set (+additional DMV pattern)		precision	recall	F_1
MALT generated patterns only		**.1769**	**.2010**	**.1882**
+p:title	$1 * $2 of	+0.73%	+8.40%	+4.14%
+p:title	$1 , * $2 of	+0.90%	+4.22%	+2.39%
+o:state_of_hqs	$1 * in * , $2	+1.35%	+1.59%	+1.43%
+p:title	$1 , * $2 who	+0.90%	+1.35%	+1.22%
+o:parents	$1 , * by $2	+0.62%	+1.35%	+1.06%
+o:city_of_hqs	$1 , * in $2 ,	+1.01%	+1.04%	+1.00%
+p:origin	$2 's $1 won the	+0.84%	+1.04%	+0.95%
+p:employee_of	$1 * $2 's chief	+0.28%	+1.04%	+0.79%
+o:website	$1 : $2	+0.28%	+1.04%	+0.79%

Table 1: DMV patterns improving MALT results the most, when added to the MALT patternset

dependency trees solely consisting of head-to-dependent edges from the right to the left[6].

All the extracted sentences were parsed and patterns were extracted from the parses. The patterns were then applied to the corpus and their precision was determined according to Freebase. With different cut-off values on training precision, the full relation extraction pipeline described in Section 2 was evaluated with respect to the Slot Filling test queries of TAC KBP 2011.

3.2 Results

Figure 2 (left) depicts F_1-measured testset results for pattern sets with varying training precision thresholds. Figure 2 (right) shows a precision recall plot of the same data points.

As can be seen in Figure 2 (left), flattening graph patterns to surface-based patterns increased the overall F_1 score. The curve for MALT generated surface patterns in Figure 2 (right) shows no increase in precision towards low recall levels where only the highest-training-precision patterns are retained. This indicates a lack of precision in MALT-based surface patterns. In contrast, the corresponding DMV-based graph increases monotonically towards lower recall levels, which is reflected by the highest F_1 score (Figure 2, left).

Table 1 shows the increases in evaluation score of those DMV-generated patterns which help most to more precisely identify relations when added to the set of all MALT-generated patterns (sorted by F_1 score). Figure 1 compares the syntactic analyses of MALT and DMV for an example sentence where DMV generates one of the listed patterns. The numbers of Table 1 indicate that such patterns are missing without alternatives in the pattern set gained from supervised parsing.

4 Conclusion

We have presented a simple method for generating surface-based patterns from parse trees which, besides avoiding the need for parsing test data, also increases extraction quality. By comparing supervised and unsupervised parsing, we furthermore found that unsupervised parsing not only eliminates the dependency on expensive domain-specific training data, but also produce surface-based extraction patterns of increased quality. Our results emphasize the need for task-driven evaluation of unsupervised parsing methods and show that there exist indicative structures for relation extraction beyond widely agreed-on linguistic syntax analyses.

5 Acknowledgements

Benjamin Roth is a recipient of the Google Europe Fellowship in Natural Language Processing, and this research is supported in part by this Google Fellowship.

[6]Since for such parses the shortest path is the complete observed word sequence between the two relation arguments, surface and parse-tree patterns become equal.

References

Enrique Alfonseca, Katja Filippova, Jean-Yves Delort, and Guillermo Garrido. 2012. Pattern learning for relation extraction with a hierarchical topic model. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 54–59, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Razvan C. Bunescu and Raymond J. Mooney. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 724–731, Stroudsburg, PA, USA. Association for Computational Linguistics.

Shay B. Cohen and Noah A. Smith. 2009. Shared logistic normal distributions for soft parameter tying in unsupervised grammar induction. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 74–82, Stroudsburg, PA, USA. Association for Computational Linguistics.

Christian Hänig and Martin Schierle. 2009. Relation extraction based on unsupervised syntactic parsing. In Gerhard Heyer, editor, *Text Mining Services*, Leipziger Beiträge zur Informatik, pages 65–70, Leipzig, Germany. Leipzig University.

William Headden. 2012. *Unsupervised Bayesian Lexicalized Dependency Grammar Induction*. Ph.D. thesis, Brown University.

James Henderson, Paola Merlo, Ivan Titov, and Gabriele Musillo. 2013. Multi-lingual joint parsing of syntactic and semantic dependencies with a latent variable model. *Computational Linguistics*, 39(4).

Heng Ji, Ralph Grishman, and Hoa Dang. 2011. Overview of the TAC2011 knowledge base population track. In *TAC 2011 Proceedings Papers*.

Dan Klein and Christopher D. Manning. 2004. Corpus-based induction of syntactic structure: models of dependency and constituency. In *ACL*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

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

Dekang Lin and Patrick Pantel. 2001. DIRT: Discovery of Inference Rules from Text. In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'01)*, pages 323–328, New York, NY, USA. ACM Press.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: the penn treebank. *Comput. Linguist.*, 19(2):313–330, June.

Bonan Min, Xiang Li, Ralph Grishman, and Sun Ang. 2012. New york university 2012 system for kbp slot filling. In *Proceedings of the Fifth Text Analysis Conference (TAC 2012)*. National Institute of Standards and Technology (NIST), November.

M. Mintz, S. Bills, R. Snow, and D. Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT)*, pages 149–160.

Hoifung Poon and Pedro Domingos. 2009. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*, EMNLP '09, pages 1–10, Stroudsburg, PA, USA. Association for Computational Linguistics.

Roi Reichart and Ari Rappoport. 2009. Automatic selection of high quality parses created by a fully unsupervised parser. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 156–164, Stroudsburg, PA, USA. Association for Computational Linguistics.

Benjamin Roth, Grzegorz Chrupala, Michael Wiegand, Singh Mittul, and Klakow Dietrich. 2012. Generalizing from freebase and patterns using cluster-based distant supervision for tac kbp slotfilling 2012. In *Proceedings of the Fifth Text Analysis Conference (TAC 2012)*, Gaithersburg, Maryland, USA, November. National Institute of Standards and Technology (NIST).

Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. 2013. A survey of noise reduction

methods for distant supervision. In *Proceedings of the 2013 workshop on Automated knowledge base construction*, pages 73–78. ACM.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2005. Learning syntactic patterns for automatic hypernym discovery. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 1297–1304. MIT Press, Cambridge, MA.

Alexander Yates and Oren Etzioni. 2009. Unsupervised methods for determining object and relation synonyms on the web. *J. Artif. Int. Res.*, 34(1):255–296, March.

Min Zhang, Jie Zhang, Jian Su, and Guodong Zhou. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 825–832, Stroudsburg, PA, USA. Association for Computational Linguistics.

Automatic Selection of Reference Pages in Wikipedia for Improving Targeted Entities Disambiguation

Takuya Makino
Fujitsu Laboratories Ltd.
4-1-1 Kamikodanaka, Nakahara-ku, Kawasaki, Japan
makino.takuya@jp.fujitsu.com

Abstract

In Targeted Entity Disambiguation setting, we take (i) a set of entity names which belong to the same domain (target entities), (ii) candidate mentions of the given entities which are texts that contain the target entities as input, and then determine which ones are true mentions of "target entity". For example, given the names of IT companies, including Apple, we determine Apple in a mention denotes an IT company or not. Prior work proposed a graph based model. This model ranks all candidate mentions based on scores which denote the degree of relevancy to target entities. Furthermore, this graph based model could utilize reference pages of target entities. However, human annotators must select reference pages in advance. We propose an automatic method that can select reference pages. We formalize the selection problem of reference pages as an Integer Linear Programming problem. We show that our model works as well as the prior work that manually selected reference pages.

1 Introduction

The enterprise is typically interested in customer's opinions. One of the methods to analyze customer's opinions is to collect mentions which contain product names. We would get a noisy mention collection if we use a simple method which extracts mentions that contain product names, since the product names may be used as other meanings.

Wang et al. (2012) proposed a new task which they referred to as Targeted Entity Disambiguation (TED). In this problem setting, we take (i) a set of entity names which belong to the same domain (target entities), (ii) candidate mentions of the given entities which are texts that contain the target entity entities as input, and then determine which ones are true mentions for the target entities. TED is different from traditional Word Sense Disambiguation or Entity Linking. Word Sense Disambiguation can be viewed as a classification task in which word senses are the classes (Navigli, 2009) and Entity Linking is the task of linking name in Web text with entities in Wikipedia (Han et al., 2011). The uniqueness of this problem is that the entities are all in the same domain (referred to as the target domain) and not necessarily included in a knowledge base such as DBpedia, Freebase or YAGO.

Wang et al. (2012) realized TED with a graph based model. In their graph based method, a target entity in a mention is regarded as a node, and the weight of an edge is determined according to context similarity, and a prior score of node that is determined according to the unique number of target entities in the mention. This graph is called as a mention graph. Using mention graph, the authority of each mention is calculated with MentionRank which is a variant of PageRank (Page et al., 1999). This authority denotes a score of how likely this node is in the target domain. In addition, MentionRank could integrate external knowledge such as Wikipedia. For each target entity, a reference page is added as a virtual node to the graph. Since reference pages can be regarded as true mentions, the prior scores of virtual nodes are higher than other mentions. This extended method can propagate the score of the virtual node of each entity to candidate mentions which are likely true. Although the use of reference pages works well, human annotators must select these reference pages.

In Word Sense Disambiguation and Entity Linking, there are some collective approaches (Hoffart et al., 2011; Kulkarni et al., 2009). In this paper, we apply this technique to the selection problem of reference pages for TED. To select refer-

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 106–110,
Gothenburg, Sweden, April 26-30 2014.

ence pages, we collect candidate reference pages of target entities from Wikipedia in advance. If the name of a target entity has a disambiguation page in Wikipedia, we have two or more candidate reference pages. Then we formalize the problem of reference page selection as an Integer Linear Programming problem. Our model is going to maximize the summation of similarities between selected pages under some constraints. Thus, coherent pages are selected as reference pages. Our method does not require any knowledge except for names of target entities. We give only target entities as input to select reference pages. Our method shows competitive accuracy of the prior method with manually selected reference pages.

2 Task Definition

Following previous work, we assume that all occurrences of a name in a mention refer to the same entity (e.g., occurrences of the string "Apple" in a single mention either all refer to the IT company or all refer to the fruit) (Wang et al., 2012).

TED is defined as follows.

Definition 1 (Targeted Entity Disambiguation)**.** *Given input of a target entity set* $E = \{e_1, ..., e_n\}$*, a mention set* $D = \{d_1, ..., d_n\}$ *and candidate mentions* $R = \{(e_i, d_j)|e_i \in E, d_j \in D\}$*, output score* $r_{ij} \in [0, 1]$ *for every candidate mention* $(e_i, d_j) \in R$*.*

3 Related Work

Wang et al. (2012) proposed MentionRank to address TED. MentionRank is similar to PageRank. This model is based on three hypotheses:

1. **Context similarity:** The true mentions across all the entities, across all the mentions will have more similar contexts than the false mentions of different entities.

2. **Co-Mention:** If multiple target entities are co-mentioned in a mention, they are likely to be true mentions.

3. **Interdependency:** If one or more mentions among the ones with similar context is deemed likely to be a true mention, they are all likely to be true mentions.

In a mention graph, a node (e_i, d_j) denotes an entity e_i in mention d_j. The weight of edge between (e_i, d_j) and (e'_i, d'_j) is denoted as $w_{ij,i'j'}$ which is a variable normalized by context similarity $\mu_{ij,i'j'}$. Context similarities are normalized to avoid "false-boost" problem. "false-boost" problem is boosting ranking score of false mentions in a false mentions group. The normalized weight of the edge is defined as follows:

$$w_{ij,i'j'} = \begin{cases} \frac{z_{ij}}{k} & \text{if} \quad i = i', \\ \frac{\mu_{i'j',ij}}{V_i Z} + \frac{z_{ij}}{k} & \text{otherwise.} \end{cases} \tag{1}$$

$$z_{ij} = 1 - \frac{\sum_{i' \neq i} \sum_{j'} \mu_{i'j',ij}}{V_i Z}, \tag{2}$$

$$Z = \max_{i,j} \frac{\sum_{i' \neq i} \sum_{j'} \mu_{i'j',ij}}{V_i}, \tag{3}$$

where, V_i denotes the number of candidate mentions that contain e_i (i.e. $V_i = |\{d_j|(e_i, d_j) \in R\}|$). k denotes the number of all candidate mentions (i.e. $k = |R|$). Co-mention is represented by a prior score. Wang et al. (2012) defined prior score π_{ij} of (e_i, d_j) as the number of unique names of target entities occurred in d_j.

The final score of each mention is decided by its prior score estimation as well as the score of the other correlated mentions.

$$r_{ij} = \lambda p_{ij} + (1 - \lambda) \sum_{i',j'} w_{ij,i'j'} r_{i'j'}, \tag{4}$$

where λ is the dumping factor. p_{ij} denotes prior score of (e_i, d_j): $p_{ij} = \pi_{ij} / \sum_{i',j'} \pi_{i'j'}$

Although this model works even if only the names of entities are given as input, we can extend this model to integrate external knowledge such as Wikipedia. For example, we can add reference pages for each entity as virtual nodes. Since we can assume that the reference page of a target entity is a true mention with a high confidence, we assign a high prior score than the other mentions. This causes the group of candidate mentions which have similar contexts with the reference pages to get higher scores. One example of using reference pages is to add a set of reference pages $\{a_i|1 \leq i \leq n\}$ into the mention graph. a_i denotes the reference page of entity e_i.

4 Proposed Method

In this section, we propose our approach for automatic selection of reference pages. In the domain of Word Sense Disambiguation and Entity Linking, some researches proposed the methods which

Figure 1: Article "Apple (disambiguation)" in Wikipedia

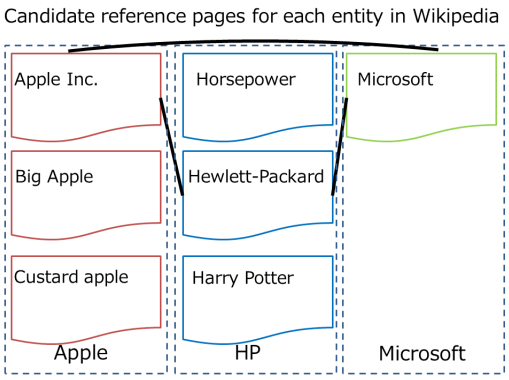


Figure 2: Automatic selection of reference pages from disambiguation pages in Wikipedia: selected pages contains same categories or similar contents (They are connected by edge).

are based on coherence between mentions (Hoffart et al., 2011; Kulkarni et al., 2009; Han et al., 2011). Our method does not require any knowledge except for the names of target entities. We give only target entities as input. Target entities in Wikipedia have two characteristics.

- A name of an ambiguous target entity tends to have a disambiguation page.
- The articles that are in the same domain have the same categories or contain similar contents.

In Wikipedia, there are disambiguation pages like Figure 1. "Apple (disambiguation)" contains apple as a plant, an IT company, a music album, and so on. To collect candidate reference pages, we use these disambiguation pages.

Kulkarni et al. (2009) formalized entity linking as an Integer Linear Programming problem and then relaxed it as a Linear Programming problem. They considered a coherence score which takes higher value if the selected articles have similar contents. Their framework can be used for entity linking and word sense disambiguation. In this paper, we use this coherence score to select reference pages. We show an image of an automatic selection of reference pages in Figure 2. In Figure 2, the target entities are Apple, HP and Microsoft. Although we have only one page for Microsoft, we have two or more candidate reference pages, since Apple and HP have disambiguation pages. Then we need to select reference pages for Apple and HP. If the name of a target entity is not in Wikipedia, we have no reference page for that target entity. The goal of this example is to select "Apple Inc." for Apple and "Hewlett-Packard" for HP (Selecting "Microsoft" for Microsoft is trivial). We regard these selected articles as reference pages for target entities.

We assume that the number of true reference page a_i for target entity e_i is one and select one reference page for each target entity. For each target entity, we select articles which the have same categories or similar contents from the set of candidate reference pages $\{c_{ik}|1 \leq k \leq l\}$ since we assume that the articles in the same domain have the same categories or contain similar contents. In fact, our model is going to maximize the summation of similarities between selected pages under some constraints. We formalize this selection as follows:

$$max. \quad \sum_{i,k} \sum_{i',k'} e_{ik,i'k'} x_{ik,i'k'},$$

$$s.t. \quad \forall i, \ \sum_{k} y_{ik} = 1, \tag{5}$$

$$y_{ik} \geq x_{ik,i'k'}; \ \forall i,k,i',k', \tag{6}$$

$$y_{i'k'} \geq x_{ik,i'k'}; \ \forall i,k,i',k', \tag{7}$$

$$x_{ik,i'k'} \in \{0,1\}; \ \forall i,k,i',k', \tag{8}$$

$$y_{ik} \in \{0,1\}; \ \forall i,k, \tag{9}$$

$e_{ik,i'k'}$ denotes the weight of the edge between candidate reference pages c_{ik} and $c_{i'k'}$. $x_{ik,i'k'}$ takes 1 if c_{ik} is selected, 0 otherwise. y_{ik} takes 1 if the edge between c_{ik} and $c_{i'k'}$ is selected, 0

	n	k	#cand	%Positive
Car	21	1809	21.5	29.9
Magazine	28	2741	17.9	43.5

Table 1: Datasets: n is # of entities, k is # of candidate mentions, #cand is average # of candidate reference pages for each entity and %Positive is % of true mentions in all candidate mentions

n=5	Car	Magazine
MentionRank	39.74	61.07
MentionRank+manVN	39.14	70.94†
MentionRank+randomVN	37.85†	65.01
Proposed method	44.21	65.86
n=10		
MentionRank	49.23	65.90†
MentionRank+manVN	47.21†	70.85
MentionRank+randomVN	45.13†	68.38
Proposed method	50.84	69.81
n=15		
MentionRank	46.50†	65.77†
MentionRank+manVN	44.29	69.38
MentionRank+randomVN	39.21†	67.89
Proposed method	42.77	69.02

Table 2: Mean average precision for each dataset

otherwise. Constraint (5) ensures that always one article is selected for each entity. Constraints (6) and (7) ensure that when $x_{ik,i'k'} = 1$, y_{ik} and $y_{i'k'}$. In this paper, we defined $e_{ik,i'k'}$ as cosine similarity of two vectors of words those weights are tfidf.

5 Experiments

We used weblogs written in Japanese for experiments. Following the previous work, we created two datasets: Car and Magazine. A summary of each dataset is shown in Table 1.

- Car: Target entities include car names such as Prius and Harrier.
- Magazine: Target entities include magazine names such as MORE and LEE.

We randomly selected 5, 10 or 15 entities from each target entities for 10 times and conducted experiment for each dataset with parameter λ = 0.15. We conducted significance test using Wilcoxon signed-rank test. Table 2 lists the experimental results on these datasets. In Table 2, MentionRank+manVN denotes MentionRank with virtual nodes that are selected manually (Wang et al., 2012). MentionRank+randomVN denotes MentionRank with virtual nodes that are selected randomly from candidate reference pages in Wikipedia. Proposed method denotes the MentionRank with virtual nodes that are selected automatically using ILP. Values with †in Table 2 indicate that there are significant differences between mean average precision of proposed method and the others. Five results of proposed methods are better than those of MentionRank, there are significant differences on two results. Furthermore, all the results of proposed method is better than those of MentionRank+randomVN and there are significant differences on three results. Four results of proposed method is worse than those of MentionRank+manVN, however there is a significant difference on only one of those results. From these results, we can see that use of reference pages automatically selected by our method improves mean average precision. In Magazine, several entities are not ambiguous and we could get true reference pages easily. Therefore, we think proposed method did not show any significant differences compared with MentionRank+randomVN. Also, in Car, several entities are not ambiguous but these reference pages belong to domains other than Car domain. As a result, we think that some results are worse than MentionRank. For example, entity "86" which is a kind of car have only one reference page that belongs to number domain.

6 Conclusion

In this paper, we proposed an automatic selection method of reference pages for Target Entity Disambiguation. Our method that uses automatically selected reference pages showed better performance than MentionRank without reference pages and competitive mean average precision with MentionRank with manually selected reference pages.

Since our framework always selects one reference page for each target entity even if a reference page does not exist in Wikipedia or one or more reference pages exist in Wikipedia, we need to refine our framework in future work. An another improvement would be to assign prior scores for virtual nodes according to coherence score between the other virtual nodes.

References

Xianpei Han, Le Sun, and Jun Zhao. 2011. Collective entity linking in web text: a graph-based method. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, SIGIR '11, pages 765–774, New York, NY, USA. ACM.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 782–792, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sayali Kulkarni, Amit Singh, Ganesh Ramakrishnan, and Soumen Chakrabarti. 2009. Collective annotation of Wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '09, pages 457–466, New York, NY, USA. ACM.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Comput. Surv.*, 41(2):10:1–10:69, February.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November.

Chi Wang, Kaushik Chakrabarti, Tao Cheng, and Surajit Chaudhuri. 2012. Targeted disambiguation of ad-hoc, homogeneous sets of named entities. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 719–728, New York, NY, USA. ACM.

Using a Random Forest Classifier to Compile Bilingual Dictionaries of Technical Terms from Comparable Corpora

Georgios Kontonatsios[1,2] **Ioannis Korkontzelos**[1,2] **Jun'ichi Tsujii**[3] **Sophia Ananiadou**[1,2]
National Centre for Text Mining, University of Manchester, Manchester, UK[1]
School of Computer Science, University of Manchester, Manchester, UK[2]
Microsoft Research Asia, Beijing, China[3]
{gkontonatsios,ikorkontzelos,sananiadou}@cs.man.ac.uk
jtsujii@microsoft.com

Abstract

We describe a machine learning approach, a *Random Forest* (RF) classifier, that is used to automatically compile bilingual dictionaries of technical terms from comparable corpora. We evaluate the RF classifier against a popular term alignment method, namely *context vectors*, and we report an improvement of the translation accuracy. As an application, we use the automatically extracted dictionary in combination with a trained Statistical Machine Translation (SMT) system to more accurately translate *unknown* terms. The dictionary extraction method described in this paper is freely available [1].

1 Background

Bilingual dictionaries of technical terms are important resources for many *Natural Language Processing* (NLP) tasks including *Statistical Machine Translation* (SMT) (Och and Ney, 2003) and *Cross-Language Information Retrieval* (Ballesteros and Croft, 1997). However, manually creating and updating such resources is an expensive process. In addition to this, new terms are constantly emerging. Especially in the biomedical domain, which is the focus of this work, there is a vast number of *neologisms*, i.e., newly coined terms, (Pustejovsky et al., 2001).

Early work on bilingual lexicon extraction focused on clean, parallel corpora providing satisfactory results (Melamed, 1997; Kay and Röscheisen, 1993). However, parallel corpora are expensive to construct and for some domains and language pairs are scarce resources. For these reasons, the focus has shifted to comparable corpora that are more readily available, more up-to-date, larger and cheaper to construct than parallel data. Comparable corpora are collections of monolingual documents in a source and target language that share the same topic, domain and/or documents are from the same period, genre and so forth.

Existing methods for bilingual lexicon extraction from comparable corpora are mainly based on the same principle. They hypothesise that a word and its translation tend to appear in similar lexical context (Fung and Yee, 1998; Rapp, 1999; Morin et al., 2007; Chiao and Zweigenbaum, 2002). Context vector methods are reported to achieve robust performance on terms that occur frequently in the corpus. Chiao and Zweigenbaum (2002) achieved a performance of 94% accuracy on the top 20 candidates when translating high frequency, medical terms (frequency of 100 or more). In contrast, Morin and Daille (2010) reported an accuracy of 21% for multi-word terms occurring 20 times or less, noting that translating rare terms is a challenging problem for context vectors.

Kontonatsios et al. (2013) introduced an RF classifier that is able to automatically learn association rules of textual units between a source and target language. However, they applied their method only on artificially constructed datasets containing an equal number of positive and negative instances. In the case of comparable corpora, the datasets are highly unbalanced (given n, m source and target terms respectively, we need to classify $n \times m$ instances). In this work, we incorporate the classification margin into the RF model, to allow the method to cope with the skewed distribution of positive and negative instances that occurs in comparable corpora.

Our proposed method ranks candidate translations using the classification margin and suggests as the best translation the candidate with the *maximum margin*. We evaluate our method on an

[1] http://personalpages.manchester.ac.uk/postgrad/georgios.kontonatsios/Software/RF-TermAlign.tar.gz

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 111–116,
Gothenburg, Sweden, April 26-30 2014.

English-Spanish comparable corpus of Wikipedia articles that are related to the medical sub-domain of "breast cancer". Furthermore, we show that dictionaries extracted from comparable corpora can be used to dynamically augment an SMT system in order to better translate *Out-of-Vocabulary (OOV)* terms.

2 Methodology

A pair of terms in a source and target language is represented as a feature vector where each dimension corresponds to a unique character n-gram. The value of each dimension is 0 or 1 and designates the occurrence of the corresponding n-gram in the input terms. The feature vectors that we use contain $2q$ dimensions where the first q dimensions correspond to the n-gram features extracted from the source terms and the last q dimensions to those from the target terms. In the reported experiments, we use the 600 (300 source and 300 target) most frequently occurring n-grams.

The underlying mechanism that allows the RF method to learn character gram mappings between terms of a source and target language is the decision trees. A node in the decision tree is a unique character n-gram. The nodes are linked through the branches of the trees and therefore the two sub-spaces of q source and q target character grams are combined. Each decision tree in the forest is constructed as follows: every node is split by considering $|\phi|$ random n-gram features of the initial feature set Ω, and a decision tree is fully grown. This process is repeated $|\tau|$ times and constructs $|\tau|$ decision trees. We tuned the RF classifier using 140 random trees where we observed a plateau in the classification performance. Furthermore, we set the number of random features using $|\phi| = \log_2 |\Omega| + 1$ as suggested by Breiman (2001).

The classification margin that we use to rank the candidate translations is calculated by simply subtracting the average number of trees predicting that the input terms are not translations from the average number of decision trees predicting that the terms are mutual translations. A larger classification margin means that more decision trees in the forest classify an instance as a translation pair.

For training an RF model, we use a bilingual dictionary of technical terms. When the dictionary lists more than one translation for an English term, we randomly select only one. Negative instances are created by randomly matching non-translation pairs of terms. We used an equal number of positive and negative instances for training the model. Starting from $20,000$ translation pairs we generated a training dataset of $40,000$ positive and negative instances.

2.1 Baseline method

The context projection method was first proposed by (Fung and Yee, 1998; Rapp, 1999) and since then different variations have been suggested (Chiao and Zweigenbaum, 2002; Morin et al., 2007; Andrade et al., 2010; Morin and Prochasson, 2011). Our implementation more closely follows the context vector method introduced by (Morin and Prochasson, 2011).

As a preprocessing step, stop words are removed using an online list [2] and lemmatisation is performed using TreeTagger (Schmid, 1994) on both the English and Spanish part of the comparable corpus. Afterwards, the method proceeds in three steps. Firstly, for each source and target term of the comparable corpus, i.e., i, we collect all lexical units that: (a) occur within a window of 3 words around i (a seven-word window) and (b) are listed in the seed bilingual dictionary. The lexical units that satisfy the above two conditions are the dimensions of the context vectors. Each dimension has a value that indicates the correlation between the context lexical unit and the term i. In our approach, we use the log-likelihood ratio. In the second step, the seed dictionary is used to translate the lexical units of the Spanish context vectors. In this way the Spanish and English vectors become comparable. When several translations are listed in the seed dictionary, we consider all of them. In the third step, we compute the *context similarity*, i.e., distance metric, between the vector of an English term to be translated with every projected, Spanish context vector. For this we use the cosine similarity.

3 Experiments

In this section, we evaluate the two dictionary extraction methods, namely context vectors and RF, on a comparable corpus of Wikipedia articles.

For the evaluation metric, we use the top-k translation accuracy [3] and the mean reciprocal

[2] http://members.unine.ch/jacques.savoy/clef/index.html

[3] the percentage of English terms whose top k candidates contain a correct translation

rank (MRR) [4] as in previous approaches (Chiao and Zweigenbaum, 2002; Chiao and Zweigenbaum, 2002; Morin and Prochasson, 2011; Morin et al., 2007; Tamura et al., 2012). As a reference list, we use the UMLS metathesaurus[5]. In addition to this, considering that in several cases the dictionary extraction methods retrieved synonymous translations that do not appear in the reference list, we manually inspected the answers. Finally, unlike previous approaches (Chiao and Zweigenbaum, 2002), we do not restrict the test list only to those English terms whose Spanish translations are known to occur in the target corpus. In such cases, the performance of dictionary extraction methods have been shown to achieve a lower performance (Tamura et al., 2012).

3.1 Data

We constructed a comparable corpus of Wikipedia articles. For this, we used Wikipedia's search engine [6] and submitted the queries "breast cancer" and "cáncer de mama" for English and Spanish respectively. From the returned list of Wikipedia pages, we used the $1,000$ top articles for both languages.

The test list contains $1,200$ English single-word terms that were extracted by considering all nouns that occur more than 10 but not more than 200 times and are listed in UMLS. For the Spanish part of the corpus, we considered all nouns as candidate translations ($32,347$ in total).

3.2 Results

Table 1 shows the top-k translation accuracy and the MRR of RF and context vectors.

	Acc_1	Acc_{10}	Acc_{20}	MRR
RF	0.41	0.57	0.59	0.47
Cont. Vectors	0.1	0.21	0.26	0.11

Table 1: top-k translation accuracy and MRR of RF and context vectors on $1,200$ English terms

We observe that the proposed RF method achieves a considerably better top-k translation accuracy and MRR than the baseline method. Moreover, we segmented the $1,200$ test terms into 7 frequency ranges [7], from high-frequency to rare terms. Figure 1 shows the translation accuracy at top 20 candidates for the two methods. We note

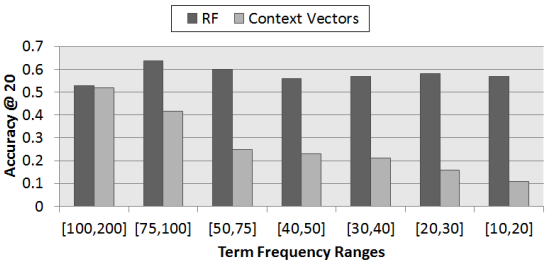


Figure 1: Translation accuracy of top 20 candidates on different frequency ranges

that for high frequency terms, i.e. [100,200] range, the performance achieved by the two methods is similar (53% and 52% for the RF and context vectors respectively). However, for lower frequency terms, the translation accuracy of the context vectors continuously declines. This confirms that context vectors do not behave robustly for rare terms (Morin and Daille, 2010). In contrast, the RF slightly fluctuates over different frequency ranges and presents approximately the same translation accuracy for both frequent and rare terms.

4 Application

As an application of our method, we use the previously extracted dictionaries to on-line augment the phrase table of an SMT system and observe the translation performance on test sentences that contain OOV terms. For the translation probabilities in the phrase table, we use the distance metric given by the dictionary extraction methods i.e., classification margin and cosine similarity of RF and context vectors respectively, normalised by the uniform probability (if a source term has m candidate translations, we normalise the distance metric by dividing by m as in (Wu et al., 2008) .

4.1 Data and tools

We construct a parallel, sentence-aligned corpus from the biomedical domain, following the process described in (Wu et al., 2011; Yepes et al., 2013). The parallel corpus comprises of article titles indexed by PubMed in both English and Spanish. We collect 120K parallel sentences for train-

[4] $MRR = \frac{1}{|Q|}\sum_{i=1}^{Q}\frac{1}{rank_i}$ where $|Q|$ is the number of English terms for which we are extracting translations and $rank_i$ is the position of the first correct translation from returned list of candidates

[5] nlm.nih.gov/research/umls

[6] http://en.wikipedia.org/wiki/Help:Searching

[7] each frequency range contains 100 randomly sampled terms

ing the SMT and 1K sentences for evaluation. The test sentences contain $1,200$ terms that do not appear in the training parallel corpus. These terms occur in the Wikipedia comparable corpus. Hence, the previously extracted dictionaries list a possible translation. Using the PubMed parallel corpus, we train Moses (Koehn et al., 2007), a phrase-based SMT system.

4.2 Results

We evaluated the translation performance of the SMT that uses the dictionary extracted by the RF against the following baselines: (i) Moses using only the training parallel data (Moses), (ii) Moses using the dictionary extracted by context vectors (Moses+context vector). The evaluation metric is BLEU (Papineni et al., 2002).

Table 2 shows the BLEU score achieved by the SMT systems when we append the top-k translations to the phrase table.

	BLEU on top-k translations		
	1	10	20
Moses	24.22	24.22	24.22
Moses+ RF	**25.32**	24.626	24.42
Moses+ Context Vectors	23.88	23.69	23.74

Table 2: Translation performance when adding top-k translations to the phrase table

We observe that the best performance is achieved by the RF when we add the top 1 translation with a total gain of 1.1 BLEU points over the baseline system. In contrast, context vectors decreased the translation performance of the SMT system. This indicates that the dictionary extracted by the context vectors is too noisy and as a result the translation performance dropped. Furthermore, it is noted that the augmented SMT systems achieve the highest performance for the top 1 translation while for k greater than 1, the translation performance decreases. This behaviour is expected since the target language model was trained only on the training Spanish sentences of the parallel corpus. Hence, the target language model does not have a prior knowledge of the OOV translations and as a result it cannot choose the correct translation among k candidates.

To further investigate the effect of the language model on the translation performance of the augmented SMT systems, we conducted an oracle experiment. In this ideal setting, we assume a strong language model, that is trained on both training and test Spanish sentences of the parallel corpus, in order to assign a higher probability to a correct translation if it exists in the deployed dictionary. As we observe in Table 3, a strong language model can more accurately select the correct translation among top-k candidates. The dictionary extracted by the RF improved the translation performance by 2.5 BLEU points for the top-10 candidates and context vectors by 0.45 for the top-20 candidates.

	BLEU on top-k translations		
	1	10	20
Moses	28.85	28.85	28.85
Moses+ RF	30.98	**31.35**	31.2
Moses+ Context Vectors	28.18	29.17	29.3

Table 3: Translation performance when adding top-k translations to the phrase table. SMT systems use a language model trained on training and test Spanish sentences of the parallel corpus.

5 Discussion

In this paper, we presented an RF classifier that is used to extract bilingual dictionaries of technical terms from comparable corpora. We evaluated our method on a comparable corpus of Wikipedia articles. The experimental results showed that our proposed method performs robustly when translating both frequent and rare terms.

As an application, we used the automatically extracted dictionary to augment the phrase table of an SMT system. The results demonstrated an improvement of the overall translation performance.

As future work, we plan to integrate the RF classifier with context vectors. Intuitively, the two methods are complementary considering that the RF exploits the internal structure of terms while context vectors use the surrounding lexical context. Therefore, it will be interesting to investigate how we can incorporate the two feature spaces in a machine learner.

6 Acknowledgements

This work was funded by the European Community's Seventh Framework Program (FP7/2007-2013) [grant number 318736 (OSSMETER)].

References

Daniel Andrade, Tetsuya Nasukawa, and Jun'ichi Tsujii. 2010. Robust measurement and comparison of context similarity for finding translation pairs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 19–27, Stroudsburg, PA, USA. Association for Computational Linguistics.

Lisa Ballesteros and W.Bruce Croft. 1997. Phrasal translation and query expansion techniques for cross-language information retrieval. In *ACM SIGIR Forum*, volume 31, pages 84–91. ACM.

Leo Breiman. 2001. Random Forests. *Machine Learning*, 45:5–32.

Yun-Chuang Chiao and Pierre Zweigenbaum. 2002. Looking for Candidate Translational Equivalents in Specialized, Comparable Corpora. In *International Conference on Computational Linguistics*.

Pascale Fung and Lo Yuen Yee. 1998. An ir approach for translating new words from nonparallel, comparable texts. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*, pages 414–420. Association for Computational Linguistics.

Martin Kay and Martin Röscheisen. 1993. Text-translation alignment. *computational Linguistics*, 19(1):121–142.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, pages 177–180. Association for Computational Linguistics.

Georgios Kontonatsios, Ioannis Korkontzelos, Jun'ichi Tsujii, and Sophia Ananiadou. 2013. Using random forest to recognise translation equivalents of biomedical terms across languages. In *Proceedings of the Sixth Workshop on Building and Using Comparable Corpora*, pages 95–104. Association for Computational Linguistics, August.

I. Dan Melamed. 1997. A portable algorithm for mapping bitext correspondence. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, pages 305–312. Association for Computational Linguistics.

Emmanuel Morin and Béatrice Daille. 2010. Compositionality and lexical alignment of multi-word terms. *Language Resources and Evaluation*, 44(1-2):79–95.

Emmanuel Morin and Emmanuel Prochasson. 2011. Bilingual lexicon extraction from comparable corpora enhanced with parallel corpora. In *Proceedings of the 4th Workshop on Building and Using Comparable Corpora: Comparable Corpora and the Web*, pages 27–34, Portland, Oregon, June. Association for Computational Linguistics.

Emmanuel Morin, Béatrice Daille, Koichi Takeuchi, and Kyo Kageura. 2007. Bilingual terminology mining - using brain, not brawn comparable corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 664–671, Prague, Czech Republic, June. Association for Computational Linguistics.

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

James Pustejovsky, Jose Castano, Brent Cochran, Maciej Kotecki, and Michael Morrell. 2001. Automatic extraction of acronym-meaning pairs from medline databases. *Studies in health technology and informatics*, (1):371–375.

Reinhard Rapp. 1999. Automatic identification of word translations from unrelated english and german corpora. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 519–526. Association for Computational Linguistics.

Helmut Schmid. 1994. Probabilistic part-of-speech tagging using decision trees. In *Proceedings of the International Conference on New Methods in Language Processing*, volume 12, pages 44–49. Manchester, UK.

Akihiro Tamura, Taro Watanabe, and Eiichiro Sumita. 2012. Bilingual lexicon extraction from comparable corpora using label propagation. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 24–36. Association for Computational Linguistics.

Hua Wu, Haifeng Wang, and Chengqing Zong. 2008. Domain adaptation for statistical machine translation with domain dictionary and monolingual corpora. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 993–1000. Association for Computational Linguistics.

Cuijun Wu, Fei Xia, Louise Deleger, and Imre Solti. 2011. Statistical machine translation for biomedical text: are we there yet? In *AMIA Annual Symposium Proceedings*, volume 2011, page 1290. American Medical Informatics Association.

Antonio Jimeno Yepes, Élise Prieur-Gaston, and Aurélie Névéol. 2013. Combining medline and publisher data to create parallel corpora for the automatic translation of biomedical text. *BMC bioinformatics*, 14(1):146.

Comparing methods for deriving intensity scores for adjectives

Josef Ruppenhofer[*], **Michael Wiegand**[†], **Jasper Brandes**[*]
[*]Hildesheim University
Hildesheim, Germany
{ruppenho|brandesj}@uni-hildesheim.de
[†]Saarland University
Saarbrücken, Germany
michael.wiegand@lsv.uni-saarland.de

Abstract

We compare several different corpus-based and lexicon-based methods for the scalar ordering of adjectives. Among them, we examine for the first time a low-resource approach based on distinctive-collexeme analysis that just requires a small predefined set of adverbial modifiers. While previous work on adjective intensity mostly assumes one single scale for all adjectives, we group adjectives into different scales which is more faithful to human perception. We also apply the methods to both polar and non-polar adjectives, showing that not all methods are equally suitable for both types of adjectives.

1 Introduction

Ordering adjectives by strength (e.g. *good* < *great* < *excellent*) is a task that has recently received much attention due to the central role of intensity classification in sentiment analysis. However, the need to assess the relative strength of adjectives also applies to non-polar adjectives. We are thus interested in establishing prior or lexical intensity scores and rankings for arbitrary sets of adjectives that evoke the same scale.[1] We do not address contextualized intensity, i.e. the fact that e.g. negation and adverbs such as *very* or *slightly* impact the perceived intensity of adjectives.

We work with four scales of adjectives (cf. Table 1). Our polar adjectives include 29 adjectives referring to quality and 18 adjectives relating to intelligence. Our non-polar adjectives include 8 dimensional adjectives denoting size and 22 denoting duration. The adjectives are taken, in part, from FrameNet's (Baker et al., 1998) frames for DESIRABILITY, MENTAL PROPERTY, SIZE and DURATION DESCRIPTION. These scales are used because they are prototypical and have multiple members on the positive and negative half-scales.

We evaluate several corpus- and resource-based methods that have been used to assign intensity scores to adjectives. We compare them to a new corpus-based method that is robust and of low complexity, and which directly uses information related to degree modification of the adjectives to be orderered. It rests on the observation that adjectives with different types of intensities co-occur with different types of adverbial modifiers.[2]

POLAR ADJECTIVES	
Intelligence Adjs.	**Intensity Level**
brilliant	very high positive
ingenious	high positive
brainy, intelligent	medium positive
smart	low positive
bright	very low positive
daft	very low negative
foolish	low negative
inane	lower medium negative
dim	upper medium negative
dim-witted, dumb, mindless	high negative
brainless, idiotic, imbecillic, moronic, stupid	very high negative
Quality Adjs.	**Intensity Level**
excellent, extraordinary, first-rate, great, outstanding, super, superb, superlative, tip-top, top-notch	very high positive
good	high positive
decent	upper medium positive
fine, fair	lower medium positive
okay, average	low positive
so-so	very low positive
mediocre	very low negative
second-rate, substandard	low negative
inferior	lower medium negative
bad, crappy, lousy, poor, third-rate	medium negative
rotten	upper medium negative
awful	high negative
shitty	very high negative
DIMENSIONAL ADJECTIVES	
Size Adjs.	**Intensity Level**
colossal, enormous, gargantuan, giant, gigantic, ginormous, humongous	high positive
big, huge, immense, large, oversize, oversized, vast	medium positive
outsize, outsized	low positive
diminutive, little, puny, small	low negative
tiny	medium negative
microscopic	high negative
Duration Adjs.	**Intensity Level**
long	high positive
lengthy	medium positive
extended	low positive
momentaneous	low negative
brief, fleeting, momentary	medium negative
short	high negative

Table 1: Adjectives used grouped by human gold standard intensity classes

[1] As there has been previous work on how to group adjectives into scales (Hatzivassiloglou and McKeown, 1993), we consider this grouping as given.

[2] The ratings we collected and our scripts are available at www.uni-hildesheim.de/ruppenhofer/data/DISA_data.zip.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 117–122,
Gothenburg, Sweden, April 26-30 2014.

2 Data and resources

Table 2 gives an overview of the different corpora and resources that we use to produce the different scores and rankings that we want to compare. The corpora and ratings will be discussed alongside the associated experimental methods in §4.1 and §4.2.

Corpora	Tokens	Reference
BNC	∼112 M	(Burnard, 2007)
LIU reviews	∼1.06 B	(Jindal and Liu, 2008)
ukWaC	∼2.25 B	(Baroni et al., 2009)
Resources	**Entries**	**Reference**
Affective norms	∼14 K	(Warriner et al., 2013)
SoCAL	∼ 6.5 K	(Taboada et al., 2011)
SentiStrength	∼ 2.5 K	(Thelwall et al., 2010)

Table 2: Corpora and resources used

3 Gold standard

We collected human ratings for our four sets of adjectives. All items were rated individually, in randomized order, under conditions that minimized bias. Participants were asked to use a horizontal slider, dragging it in the desired direction, representing polarity, and releasing the mouse at the desired intensity, ranging from -100 to $+100$.

Through Amazon Mechanical Turk (AMT), we recruited subjects with the following qualifications: US residency, a HIT-approval rate of at least 96% (following Akkaya et al. (2010)), and 500 prior completed HITs. We collected 20 ratings for each item but had to exclude some participants' answers as unusable, which reduced our sample to 17 subjects for some items. In the raw data, all adjectives had different mean ratings and their standard deviations overlapped. We therefore transformed the data into sets of equally strong adjectives as follows. For a given pair of adjectives of identical polarity, we counted how many participants rated adjective A more intense than adjective B; B more intense than A; or A as intense as B. Whenever a simple majority existed for one of the two unequal relations, we adopted that as our relative ranking for the two adjectives.[3] The resulting rankings (intensity levels) are shown in Table 1.

4 Methods

Our methods to determine the intensity of adjectives are either corpus- or lexicon-based.

[3]In our data, there was no need to break circular rankings, so we do not consider this issue here.

4.1 Corpus-based methods

Our first method, **distinctive-collexeme analysis (Collex)** (Gries and Stefanowitsch, 2004) assumes that adjectives with different types of intensities co-occur with different types of adverbial modifiers (Table 3). End-of-scale modifiers such as *extremely* or *absolutely* target adjectives with a partially or fully closed scale, such as *brilliant* or *outstanding*, which occupy extreme positions on the intensity scale. "Normal" degree modifiers such as *very* or *rather* target adjectives with an open scale structure (in the sense of Kennedy and McNally (2005)), such as *good* or *decent*, which occupy non-extreme positions.

To determine an adjective's preference for one of the two constructions, the Fisher exact test (Pedersen, 1996) is used. It makes no distributional assumptions and does not require a minimum sample size. The direction in which observed values differ from expected ones indicates a preference for one construction over the other and the p-values are taken as a measure of the preference strength. Our hypothesis is that e.g. an adjective A with greater preference for the end-of-scale construction than adjective B has a greater inherent intensity than B. We ran distinctive-collexeme analysis on both the ukWaC and the BNC. We refer to the output as **Collex**$_{ukWaC}$ and **Collex**$_{BNC}$. Note that this kind of method has ***not*** yet been examined for automatic intensity classification.

end-of-scale	"normal"
100%, fully, **totally**, absolutely, **completely**, perfectly, entirely, utterly, **almost**, partially, half, mostly	all, as, awfully, enough, extremely, fairly, highly, how, least, less, much, pretty, quite, rather, **so**, somewhat, sort of, terribly, **too**, **very**, well

Table 3: Domain independent degree modifiers (3 most freq. terms in the **BNC**; 3 most freq. terms in the ukWaC)

Another corpus-based method we consider employs **Mean star ratings (MeanStar)** from product reviews as described by Rill et al. (2012). Unlike Collex, this method uses no linguistic properties of the adjectives themselves. Instead, it derives intensity from the star rating scores that reviewers (manually) assign to reviews. We count how many instances of each adjective i (of the set of adjectives to classify) occur in review titles with a given star rating (score) S_j within a review corpus. The intensity score is defined as the weighted mean of the star ratings $SR_i = \frac{\sum_{j=1}^{n} S_j^i}{n}$.

Horn (1976) proposes **pattern-based diagnos-**

Pattern	Any	Int.	Qual.	Size	Dur.
X or even Y	4118	1	34	9	3
X if not Y	3115	1	0	29	0
be X but not Y	2815	0	74	3	1
not only X but Y	1114	0	3	0	0
X and in fact Y	45	0	0	0	0
not X, let alone Y	4	0	0	0	0
not Y, not even X	4	0	1	0	0

Table 4: Phrasal patterns in the ukWaC

tics for acquiring information about the scalar structure of adjectives. This was validated on actual data by Sheinman and Tokunaga (2009). A pattern such as *not just/only X but Y* implies that [Y] must always be stronger than [X] (as in *It's not just good but great.*).

The pattern-based approach has a *severe* coverage problem. Table 4 shows the results for 7 common phrasal patterns in the larger of our two corpora, the ukWaC. The slots in the patterns are typically *not* filled by adjectives from the same scale. For example, the most frequent pattern *X or even Y* has 4118 instances in the ukWaC. Only 34 of these have quality adjectives in both slots. Though de Melo and Bansal (2013) have shown that the coverage problems can be overcome and state-of-the-art results obtained using web scale data in the form of Google n-grams, we still set aside this method here because of its great resource need.

4.2 Manually compiled lexical resources

In addition to the corpus methods, we also consider some manually compiled resources. We want to know if the polarity and intensity information in them can be used for ordering polar adjectives.

One resource we consider are the affective ratings (elicited with AMT) for almost 14,000 English words collected by Warriner et al. (2013). They include scores of valence (*unhappy* to *happy*), arousal (*calm* to *aroused*) and dominance (*in control* to *controlled*) for each word in the list. This scoring system follows the dimensional theory of emotion by Osgood et al. (1957). We will interpret each of these dimensions as a separate intensity score, i.e. **War**$_{Val}$, **War**$_{Aro}$ and **War**$_{Dom}$.

Beyond Warriner's ratings, we consider the two polarity lexicons **SentiStrength** (Thelwall et al., 2010) and **SoCAL** (Taboada et al., 2011) which also assign intensity scores to polar expressions.

5 Experiments

For our evaluation, we compute the similarity between the gold standard and every other ranking we are interested in in terms of Spearman's rank correlation coefficient (Spearman's ρ).

	Polar		Dimensional	
Data set	Intelligence	Quality	Duration	Size
MeanStar	0.886	0.935	0.148	-0.058
SoCAL	0.848	0.953	NA	0.776
SentiStrength	0.874	0.880	NA	NA
Collex$_{ukWaC}$	0.837	0.806	0.732	0.808
Collex$_{ukWaC*}$	0.845	0.753	0.732	0.940
Collex$_{BNC}$	0.834	0.790	0.732	0.733
Collex$_{BNC*}$	0.705	0.643	0.834	0.700
War$_{Val}$	0.779	0.916	-0.632	-0.031
War$_{Aro}$	0.504	-0.452	0.316	0.717
War$_{Dom}$	0.790	0.891	0.632	0.285

Table 5: Spearman rank correlations with the human gold standard (*: only the 3 most frequent modifiers are used (see Table 3))

5.1 Data transformation

For the word lists with numeric scores (MeanStar (§4.1); SentiStrength, SoCAL, War$_{Val}$, War$_{Aro}$ and War$_{Dom}$ (§4.2)) we did as follows: Adjectives not covered by the word lists were ignored. Adjectives with equal scores were given tied ranks.

For the experiments involving distinctive collexeme analysis in our two corpora (§4.1) we proceeded as follows: The adjectives classified as distinctive for the end-of-scale modification constructions were put at the top and bottom of the ranking according to polarity; the greater the collostructional strength for the adjective as denoted by the p-value, the nearer it is placed to the top or bottom of the ranking. The adjectives that are distinctive for the normal degree modification construction are placed between those adjectives distinctive for the end-of-scale modification construction, again taking polarity and collostructional strength into account. This time, the least distinctive lemmas for the normal modification construction come to directly join up with the least distinctive lemmas for the end-of-scale construction. In between the normal modifiers, we place adjectives that have no preference for one or the other construction, which may result from non-occurrence in small data sets (see §5.2).

5.2 Results

The results of the pairwise correlations between the human-elicited gold standard and the rankings derived from various methods and resources are shown in Table 5. For polar adjectives, most rankings correlate fairly well with human judgments. Warriner's arousal list, however, performs poorly on quality adjectives, whereas MeanStar and Warriner's dominance and valence lists perform better on quality than on intelligence adjectives. For MeanStar, this does not come as a surprise as quality adjectives are much more frequent in prod-

uct reviews than intelligence adjectives. Overall, it seems that MeanStar most closely matches the human judgments that we elicited for the intelligence adjectives. SentiStrength also produces high scores. However, we do not have full confidence in that result since SentiStrength lacks many of our adjectives, thus leading to a possibly higher correlation than would have been achieved if ranks (scores) had been available for all adjectives.

The picture is very different for the dimensional (non-polar) adjectives. While Collex still gives very good results, especially on the ukWaC, the MeanStar method and most Warriner lists produce very low positive or even negative correlations. This shows that estimating the intensity of non-polar adjectives from metadata or ratings elicited in terms of affect is not useful. It is much better to consider their actual linguistic behavior in degree constructions, which Collex does. SentiStrength has no coverage for size or duration adjectives. SoCAL covers 14 of the 22 size adjectives.

Although it never gives the best result, Collex produces stable results across both corpora and the four scales. It also requires the least human effort by far. While all other rankings are produced with the help of heavy human annotation (even MeanStar is completely dependent on manually assigned review scores), one has only to specify some *domain-independent* degree and end-of-scale modifiers. Table 5 also shows that normally a larger set of modifiers is necessary: only considering the 3 most frequent terms (Table 3) results in a notably reduced correlation. As there is no consistent significant difference between Collex_{BNC} and Collex_{ukWaC} even though the ukWaC is 20 times larger than the BNC (Table 2), we may conclude that the smaller size of the BNC is already sufficient. This, however, raises the question whether even smaller amounts of data than the full BNC could already produce a reasonable intensity ranking. Figure 1 plots the Spearman correlation for our adjectives using various sizes of the BNC corpus.[4] It shows that further reducing the size of the corpus causes some deterioration, most significantly on the intelligence adjectives. The counter-intuitive curve for duration adjectives is explained as follows. Collex produces ties in the middle of the scale when data is lacking (see §5.1). Because the smallest corpus slices contain no or very few instances and because the gold standard does include several ties, the results for duration adjectives are inflated initially, when data is lacking.

[4]For each size, we average across 10 samples.

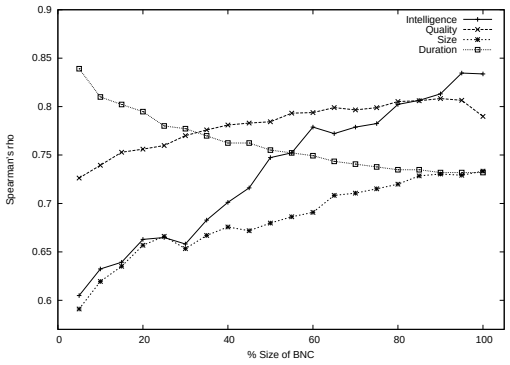


Figure 1: Reducing the size of the BNC

6 Related work

Sentiment analysis on adjectives has been extensively explored in previous work, however, most work focussed on the extraction of subjective adjectives (Wiebe, 2000; Vegnaduzzo, 2004; Wiegand et al., 2013) or on the detection of polar orientation (Hatzivassiloglou and McKeown, 1997; Kamps et al., 2004; Fahrni and Klenner, 2008).

Intensity can be considered in two ways, as a contextual strength analysis (Wilson et al., 2004) or as an out-of-context analysis, as in this paper.

Our main contribution is that we compare several classification methods that include a new effective method based on distinctive-collexeme analysis requiring hardly any human guidance and which moreover can solve the problem of intensity assignment for all, not only polar adjectives.

7 Conclusion

We compared diverse corpus-based and lexicon-based methods for the intensity classification of adjectives. Among them, we examined for the first time an approach based on distinctive-collexeme analysis. It requires only a small predefined set of adverbial modifiers and relies only on information about individual adjectives rather than co-occurrences of adjectives within patterns. As a result, it can be used with far less data than e.g. the Google n-grams provide. Unlike the mean star approach, it needs no extrinsic meta-data and it can handle both polar and non-polar adjectives. Accordingly, it appears to be very promising for cases where only few resources are available and as a source of evidence to be used in hybrid methods.

Acknowledgments

Michael Wiegand was funded by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IC12SO1X. The authors would like to thank Maite Taboada for providing her sentiment lexicon (SoCAL) to be used for the experiments presented in this paper.

References

Cem Akkaya, Alexander Conrad, Janyce Wiebe, and Rada Mihalcea. 2010. Amazon Mechanical Turk for Subjectivity Word Sense Disambiguation. In *NAACL-HLT 2010 Workshop on Creating Speech and Language Data With Amazon's Mechanical Turk*, pages 195–203, Los Angeles, CA, USA.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley Framenet Project. In *Proceedings of the International Conference on Computational Linguistics and Annual Meeting of the Association for Computational Linguistics (COLING/ACL)*, pages 86–90, Montréal, Quebec, Canada.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetti. 2009. The WaCky Wide Web: A Collection of Very Large Linguistically Processed Web-Crawled Corpora. *Language Resources and Evaluation*, 43(3):209–226.

Lou Burnard, 2007. *Reference Guide for the British National Corpus*. Research Technologies Service at Oxford University Computing Services, Oxford, UK.

Gerard de Melo and Mohit Bansal. 2013. Good, Great, Excellent: Global Inference of Semantic Intensities. *Transactions of the Association for Computational Linguistics*, 1:279–290.

Angela Fahrni and Manfred Klenner. 2008. Old Wine or Warm Beer: Target Specific Sentiment Analysis of Adjectives. In *Proceedings of the Symposium on Affective Language in Human and Machine*, pages 60–63, Aberdeen, Scotland, UK.

Stefan Th. Gries and Anatol Stefanowitsch. 2004. Extending collostructional analysis: a corpus-based perspective on 'alternations'. *International Journal of Corpus Linguistics*, 9(1):97–129.

Vasileios Hatzivassiloglou and Kathleen McKeown. 1993. Towards the Automatic Identification of Adjectival Scales: Clustering Adjectives According to Meaning. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 172–182, Columbus, OH, USA.

Vasileios Hatzivassiloglou and Kathleen R. McKeown. 1997. Predicting the Semantic Orientation of Adjectives. In *Proceedings of the Conference on European Chapter of the Association for Computational Linguistics (EACL)*, pages 174–181, Madrid, Spain.

Laurence Robert Horn. 1976. *On the Semantic Properties of Logical Operators in English*. Indiana University Linguistics Club.

Nitin Jindal and Bing Liu. 2008. Opinion Spam and Analysis. In *Proceedings of the international conference on Web search and web data mining (WSDM)*, pages 219–230, Palo Alto, USA.

Jaap Kamps, M.J. Marx, Robert J. Mokken, and Maarten De Rijke. 2004. Using Wordnet to Measure Semantic Orientations of Adjectives. In *Proceedings of the Conference on Language Resources and Evaluation (LREC)*, pages 1115–1118, Lisbon, Portugal.

Christopher Kennedy and Louise McNally. 2005. Scale Structure, Degree Modification, and the Semantics of Gradable Predicates. *Language*, 81(2):345–338.

Charles E. Osgood, George Suci, and Percy Tannenbaum. 1957. *The Measurement of Meaning*. University of Illinois Press.

Ted Pedersen. 1996. Fishing for exactness. In *Proceedings of the South-Central SAS Users Group Conference*, Austin, TX, USA.

Sven Rill, Johannes Drescher, Dirk Reinel, Joerg Scheidt, Oliver Schuetz, Florian Wogenstein, and Daniel Simon. 2012. A Generic Approach to Generate Opinion Lists of Phrases for Opinion Mining Applications. In *Proceedings of the KDD-Workshop on Issues of Sentiment Discovery and Opinion Mining (WISDOM)*, Beijing, China.

Vera Sheinman and Takenobu Tokunaga. 2009. AdjScales: Differentiating between Similar Adjectives for Language Learners. *CSEDU*, 1:229–235.

Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-Based Methods for Sentiment Analysis. *Computational Linguistics*, 37(2):267 – 307.

Mike Thelwall, Kevan Buckley, Georgios Paltoglou, and Di Cai. 2010. Sentiment Strength Detection in Short Informal Text. *Journal of the American Society for Information Science and Technology*, 61(12):2544–2558.

Stefano Vegnaduzzo. 2004. Acquisition of Subjective Adjectives with Limited Resources. In *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*, Stanford, CA, USA.

Amy Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior Research Methods*, Online First:1–17.

Janyce M. Wiebe. 2000. Learning Subjective Adjectives from Corpora. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 735–740, Austin, TX, USA.

Michael Wiegand, Josef Ruppenhofer, and Dietrich Klakow. 2013. Predicative Adjectives: An Unsupervised Criterion to Extract Subjective Adjectives. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL (HLT/NAACL)*, pages 534–539, Atlanta, GA, USA.

Theresa Wilson, Janyce Wiebe, and Rebecca Hwa. 2004. Just how mad are you? Finding strong and weak opinion clauses. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, pages 761–767, San Jose, CA, USA.

Bayesian Word Alignment for Massively Parallel Texts

Robert Östling
Department of Linguistics
Stockholm University
robert@ling.su.se

Abstract

There has been a great amount of work done in the field of bitext alignment, but the problem of aligning words in massively parallel texts with hundreds or thousands of languages is largely unexplored. While the basic task is similar, there are also important differences in purpose, method and evaluation between the problems. In this work, I present a non-parametric Bayesian model that can be used for simultaneous word alignment in massively parallel corpora. This method is evaluated on a corpus containing 1144 translations of the New Testament.

1 Introduction

Bitext word alignment is the problem of finding links between words given pairs of translated sentences (Tiedemann, 2011). Initially, this was motivated by Statistical Machine Translation (SMT) applications (Brown et al., 1993), but word-aligned texts have also been used to transfer linguistic annotation between languages (Yarowsky et al., 2001; Täckström, 2013), for Word Sense Disambiguation (WSD) (Diab and Resnik, 2002) and lexicon extraction (Wu and Xia, 1994).

Massively parallel texts, in the sense used by Cysouw and Wälchli (2007), are essentially the same as bitexts, only with hundreds or thousands of languages rather than just two. Parallel corpora used in SMT, for instance the Europarl Corpus (Koehn, 2005), tend to contain few (up to tens of) languages, but many (up to billions of) words in each language. Massively parallel corpora, on the other hand, contain many (hundreds of) languages, but usually fewer (less than a million) words in each language.

Additionally, aligned massively parallel corpora have different applications than traditional parallel corpora with pairwise alignments. Whereas the latter tend to be used for the various NLP tasks mentioned above, massively parallel corpora have mostly been used for investigations in linguistic typology (Cysouw and Wälchli, 2007).

There has been surprisingly few studies on multilingual word alignment. Mayer and Cysouw (2012) treat alignment as a clustering problem, where the words in each sentence are clustered according to some measure of co-occurrence. They provide no evaluation, but alignment methods based on co-occurrence statistics have been found to have lower accuracy than even very simple generative models (Och and Ney, 2003), so this might not be a promising direction as far as accuracy is concerned.

A related line of research is due to Lardilleux et al. (2011), who learn sets of multilingual translation equivalent phrases. Although later work (Lardilleux et al., 2013) uses phrase pairs extracted with this method for (bitext) word alignment, their method solves a somewhat different problem from what is considered here.

Some authors have studied how multilingual parallel corpora can be used to improve bitext alignment. Filali and Bilmes (2005) use (bitext) alignments to addditional languages as features in bitext alignment, while Kumar et al. (2007) interpolate alignments through multiple bridge languages to produce a bitext alignment for another language pair. Since the goal of this research is not multilingual alignment, it will not be considered further here.

2 Multilingual Alignment

In bitext alignment, the goal is to find links between individual word tokens in parallel sentence pairs. The IBM models (Brown et al., 1993) formalize this in a directional fashion where each word j in a *source language* is linked to word i in the *target language* through alignment variables $i = a_j$, thus specifying a 1-to-n mapping from

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 123–127,
Gothenburg, Sweden, April 26-30 2014.

source language words to target language words.

An intuitively appealing way to formalize the multilingual alignment problem is through a *common representation* (or *interlingua*) to which each individual language is aligned. If the common representation is isomorphic to one of the languages in the corpus, this is equivalent to using that language as a bridge. However, since all languages (and all translations) have their own idiosyncrasies that make linking to other translations difficult, it seems better to learn a common representation that corresponds to information in a sentence that is present in as many of the translations as possible.

3 Method

Recently, it has been shown that Bayesian methods that use priors to bias towards linguistically more plausible solutions can improve bitext word alignment (Mermer and Saraçlar, 2011; Riley and Gildea, 2012; Gal and Blunsom, 2013). Given these promising results and the fact that massively parallel texts tend to be rather short, which makes the role of realistic priors more important, I have decided to use a Bayesian alignment model for this work.

3.1 Model

The model used in this work uses a common representation of *concepts* generated by a Chinese Restaurant Process (CRP), which is aligned to each of the languages in a corpus using the model of Mermer and Saraçlar (2011).

Table 1 introduces the variables (observed and latent) as well as the hyperparameters of the model. Basically, the model consists of a common representation c (where token i of sentence s is denoted c_{si}), which is aligned to one or more words w_{lsj} (from language l, sentence s, token j) through a set of alignment variables a_{lsj} which contain the index within c_s that w_{lsj} is linked to.

The probability of an assignment c is:

$$\mathrm{CRP}(c;\alpha) = \frac{\Gamma(1+\alpha)}{\Gamma(n+\alpha)} \cdot \alpha^{|E_c|-1} \cdot \prod_{e \in E_c} (n_e - 1)!$$

where n_e is the number of occurrences of concept type e in the assignment c, and $n = \sum_e n_e$ is the (fixed) total number of tokens in the common representation.

For the translation probabilities, I follow Mermer and Saraçlar (2011) in assuming that $p(f_l|e) \sim \mathrm{Dir}(t_l;\theta_l)$, and that the priors θ_l are *symmetric* (i.e. all values in these vectors are equal, $\theta_{lef} = \beta$). By specifying a low value for β (a *sparse* prior), we can encode our prior knowledge that translation probability functions $p(f_l|e)$ tend to have a low entropy, or in other words, that each concept is typically only translated into a very small number of words in each language.

The joint probability of the common representation and the alignments is given by:

$$\begin{aligned} &p(c,a,w,t;\alpha,\theta) = \\ &\quad p(c;\alpha) \cdot p(w|c,a,t) \cdot p(a|c) \cdot p(t;\theta) \end{aligned} \tag{1}$$

where $p(c;\alpha) = \mathrm{CRP}(c;\alpha)$ and the remaining factors are the same as in Mermer and Saraçlar (2011) with the common representation being the "target language", except that there is a product across all languages l. Note that since word order is not modeled, $p(a|c)$ is constant.

3.2 Learning

The model is trained using a collapsed Gibbs sampler. Due to space limitations, the full derivation is omitted, but the sampling distribution turns out to be as follows for the common representation:

$$\begin{aligned} p(c_{si} = e') \propto \frac{1}{n-1+\alpha} \cdot \begin{cases} \alpha & \text{if } n_{e'} = 1 \\ n_{e'} - 1 & \text{if } n_{e'} > 1 \end{cases} \\ \cdot \prod_l \frac{\prod_{f \in A_{lsi}} \prod_{k=1}^{m_{lsif}} \left(n_{le'f} + \theta_{le'f} - k\right)}{\prod_{k=1}^{\sum_f m_{lsif}} \left(\sum_{f \in F_l} n_{le'f} + \theta_{le'f} - k\right)} \end{aligned} \tag{2}$$

where A_{lsi} is the set of word types f in language l which are aligned to c_{si}, and m_{lsif} is the number of times each such f is aligned to c_{si}. In order to speed up calculations, the product in Equation 2 can be approximated by letting l run over a small random subset of languages. The experiments carried out in this work only use this approximation when the full corpus of 1144 translations is used, then a subset of 24 languages is randomly selected when each c_{si} is sampled. An empirical evaluation of the effects of this approximation is left for future work.

The alignment sampling distribution is:

$$p(a_{lsj} = i) \propto \frac{n_{le'f'} + \theta_{le'f'} - 1}{\sum_f \left(n_{le'f} + \theta_{le'f}\right) - 1} \tag{3}$$

where $e' = c_{s a_{lsj}}$ is the concept type aligned to word type $f' = w_{lsj}$.

Rather than sampling directly from the distributions above, one can sample from $\hat{p}(c_{si} = e') \propto$

Table 1: Variables used in the model.

Observed variables	
F_l	the set of word types in language l
$w_{lsj} \in F_l$	word j of sentence s in language l
$I_s \in \mathbb{N}$	length of sentence s in the common representation
$J_{ls} \in \mathbb{N}$	length of sentence s in language l
Latent variables	
E_c	the set of concepts in the assignment c
$c_{si} \in E_c$	concept i of sentence s in the common representation
$a_{lsj} \in \{1..I_s\}$	alignment of w_{lsj} to c_{si}; $i = a_{lsj}$
$t_{lef} \in \mathbb{R}$	translation probability $p(f_l\|e)$, where $f_l \in F_l$ and $e \in E_c$
Hyperparameters	
α	CRP hyperparameter, fixed to 1000 in the experiments
β	symmetric Dirichlet prior for translation distributions $\theta_\mathbf{l}$, fixed to 0.001 in the experiments

$p(c_{si} = e')^\lambda$ and $\hat{p}(a_{lsj} = i) \propto p(a_{lsj} = i)^\lambda$. The temperature parameter λ can be varied during training to change the amount of randomness while sampling.

3.3 Initialization

In order to obtain a reasonable initial state for the Gibbs sampling, one can simply initialize the common representation to be identical to one of the languages in the corpus. For this language one then (trivially) has a perfect alignment, while the remaining languages are initialized randomly and their alignments are learned. Random initialization of the common representation is possible, but turns out to perform poorly.

4 Experiments

The most basic question about the present model is whether sampling the common representation is helpful, compared to simply choosing a language and aligning all other languages to that one.

In order to test this, I initialize the model as described in section 3.3 and sample alignments (but not the common representation) for 200 iterations with λ linearly increasing from 0 to 2, followed by two iterations with $\lambda \to \infty$. This gives a strong baseline, from which one can start learning the joint model.

4.1 Data

I use a corpus containing verse-aligned translations of the New Testament into a great number of languages. After some exclusions due to e.g. non-standard formatting or improperly segmented text, the version used in this work contains 1144 translations in 986 different languages. The mean number of tokens among the translations is 236 000, and the mean number of types is 9 500.

4.2 Evaluation Measures

Previous authors have tended to avoid multilingual evaluation altogether. Mayer and Cysouw (2012) do not evaluate their method, while Lardilleux et al. (2011) only use bilingual evaluation.

Cysouw et al. (2007) use the fact that some translations of the Bible have been annotated with Strong's Numbers, which map most word tokens to the lemma of its translation equivalent in the original language, to perform bilingual evaluation of Bible corpus alignments.

Strong's Numbers can be used in a different way to evaluate the type of multilingual alignment produced by the method in this work. Both the Strong's Numbers and the common representation can be interpreted as clusterings of the word tokens in each language. Ideally one would want these two clusterings to be identical, as they would be if the original language had been perfectly constructed. Standard clustering evaluation measures can be used for this task, and in this work I use normalized mutual information (also reinvented as *V-measure* by Rosenberg and Hirschberg (2007)). The evaluation is limited to words which are assigned exactly one Strong's Number, in an attempt to avoid some of the problems with scope discussed by Cysouw et al. (2007). Note that even a perfect alignment from one language to itself does not achieve the maximum score using this mea-

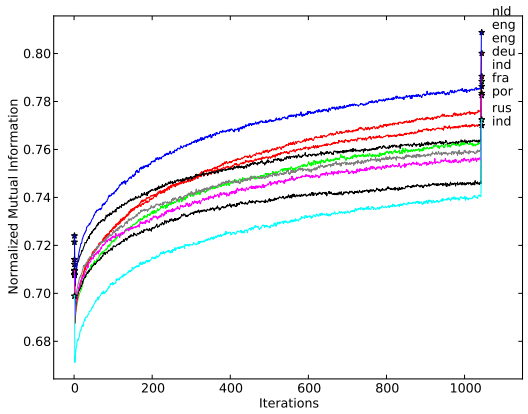


Figure 1: Alignment quality of Mandarin-initialized model.

sure, only a successful reconstruction of the original text (minus inflections) would.

In the Bible corpus used here, nine translations in seven languages contain Strong's Numbers annotations: English and Indonesian (two translations each), as well as German, French, Dutch, Portuguese and Russian (one translation each).

4.3 Results

Figure 1 shows alignment quality during training in a model initialized using a translation in Mandarin, which is not related to any of the languages in the evaluation sample and was chosen to avoid initialization bias. After an initial drop when noise is introduced during the Gibbs sampling process, alignment quality quickly increases as the common representation moves towards the versions in the evaluation sample. The final two iterations (with $\lambda \to \infty$) remove the sampling noise and the model rapidly converges to a local maximum, resulting in a sharp increase in alignment quality at the end. Further iterations only result in minor improvements.

Table 2 contains the baseline and joint model results for models initialized with either English or Mandarin versions. The joint model outperforms the baseline in all cases except when the initialization language is the same as the evaluation language (the two English translations in the left column), which is expected since it is easy to align a text to itself or to a very similar version.

The two models described so far only use the nine-translation evaluation sample to learn the common representation, since using additional languages would unfairly penalize the joint learning model. I have also tested the model on the full corpus of 1144 translations with an English-initialized model and the same training setup as above (initialized from English). In this case, alignment quality decreased somewhat for the languages most similar to English, which is to be expected since the majority of languages in the corpus are unrelated to English and pull the common representation away from the European languages in the evaluation sample. Although it is not possible to directly evaluate alignment quality outside the evaluation sample with Strong's Numbers, the log-probability of the entire data under the model (Equation 1) increases as expected, by about 5%.

	English		Mandarin	
	A	A+J	A	A+J
deu	0.817	**0.824**	0.708	**0.788**
eng	**0.854**	0.851	0.714	**0.800**
eng_2	**0.834**	0.833	0.708	**0.790**
fra	0.807	**0.816**	0.712	**0.783**
ind	0.774	**0.785**	0.710	**0.770**
ind_2	0.791	**0.803**	0.721	**0.786**
nld	0.839	**0.850**	0.724	**0.809**
por	0.807	**0.813**	0.709	**0.782**
rus	0.792	**0.800**	0.699	**0.772**

Table 2: Normalized mutual information with respect to Strong's Numbers, using alignment only (A) or joint alignment + common representation learning (A+J), for models initialized using English or Mandarin.

5 Conclusions and Future Work

As the number of translations in a parallel corpus increases, the problem of aligning them becomes a rather different one from aligning translation *pairs*. I have presented a Bayesian method that jointly learns a common structure along with alignments to each language in the corpus. In an empirical evaluation, the joint method outperforms the baseline where the common structure is one of the languages.

Currently the underlying alignment model is quite simplistic, and preliminary results indicate that including the HMM word order model of Vogel et al. (1996) further improves alignments.

Acknowledgments

Thanks to Jörg Tiedemann, Mats Wirén and the anonymous reviewers for their comments.

References

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

Michael Cysouw and Bernhard Wälchli. 2007. Parallel texts: Using translational equivalents in linguistic typology. *STUF - Language Typology and Universals*, 60(2):95–99.

Michael Cysouw, Chris Biemann, and Matthias Ongyerth. 2007. Using Strong's Numbers in the Bible to test an automatic alignment of parallel texts. *STUF - Language Typology and Universals*, 60(2):158–171.

Mona Diab and Philip Resnik. 2002. An unsupervised method for word sense tagging using parallel corpora. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 255–262, Stroudsburg, PA, USA. Association for Computational Linguistics.

Karim Filali and Jeff Bilmes. 2005. Leveraging multiple languages to improve statistical MT word alignments. In *IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 92–97, San Juan, November. IEEE.

Yarin Gal and Phil Blunsom. 2013. A systematic bayesian treatment of the ibm alignment models. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Stroudsburg, PA, USA. Association for Computational Linguistics.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *The Tenth Machine Translation Summit*, Phuket, Thailand.

Shankar Kumar, Franz J. Och, and Wolfgang Macherey. 2007. Improving word alignment with bridge languages. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 42–50, Prague, Czech Republic, June. Association for Computational Linguistics.

Adrien Lardilleux, Yves Lepage, and Franois Yvon. 2011. The contribution of low frequencies to multilingual sub-sentential alignment: a differential associative approach. *International Journal of Advanced Intelligence*, 3(2):189–217.

Adrien Lardilleux, Francois Yvon, and Yves Lepage. 2013. Hierarchical sub-sentential alignment with Anymalign. In *Proceedings of the 16th EAMT Conference*, pages 279–286, Trento, Italy, 28-30 May 2012.

Thomas Mayer and Michael Cysouw. 2012. Language comparison through sparse multilingual word alignment. In *Proceedings of the EACL 2012 Joint Workshop of LINGVIS & UNCLH*, EACL 2012, pages 54–62, Stroudsburg, PA, USA. Association for Computational Linguistics.

Coşkun Mermer and Murat Saraçlar. 2011. Bayesian word alignment for statistical machine translation. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 182–187, Stroudsburg, PA, USA. Association for Computational Linguistics.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51, March.

Darcey Riley and Daniel Gildea. 2012. Improving the IBM alignment models using variational Bayes. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 306–310, Stroudsburg, PA, USA. Association for Computational Linguistics.

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 410–420, Prague, Czech Republic, June. Association for Computational Linguistics.

Oscar Täckström. 2013. *Predicting Linguistic Structure with Incomplete and Cross-Lingual Supervision*. Ph.D. thesis, Uppsala University, Department of Linguistics and Philology.

Jörg Tiedemann. 2011. *Bitext Alignment*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th Conference on Computational Linguistics - Volume 2*, COLING '96, pages 836–841, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dekai Wu and Xuanyin Xia. 1994. Learning an English-Chinese lexicon from a parallel corpus. In *Proceedings of the First Conference of the Association for Machine Translation in the Americas*, pages 206–213.

David Yarowsky, Grace Ngai, and Richard Wicentowski. 2001. Inducing multilingual text analysis tools via robust projection across aligned corpora. In *Proceedings of the First International Conference on Human Language Technology Research*, HLT '01, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Acquiring a Dictionary of Emotion-Provoking Events

Hoa Trong Vu[†,‡], Graham Neubig[†], Sakriani Sakti[†], Tomoki Toda[†], Satoshi Nakamura[†]
[†]Graduate School of Information Science, Nara Institute of Science and Technology
8916-5 Takayama-cho, Ikoma-shi, Nara, Japan
[‡]Vietnam National University, University of Engineering and Technology
E3 Building - 144 Xuan Thuy Street, Cau Giay, Hanoi, Vietnam

Abstract

This paper is concerned with the discovery and aggregation of events that provoke a particular emotion in the person who experiences them, or *emotion-provoking events*. We first describe the creation of a small manually-constructed dictionary of events through a survey of 30 subjects. Next, we describe first attempts at automatically acquiring and aggregating these events from web data, with a baseline from previous work and some simple extensions using seed expansion and clustering. Finally, we propose several evaluation measures for evaluating the automatically acquired events, and perform an evaluation of the effectiveness of automatic event extraction.

1 Introduction

"You look happy today, did something good happen?" This is a natural question in human dialogue, and most humans could think of a variety of answers, such as "I met my friends" or "I passed a test." In this work, we concern ourselves with creating resources that answer this very question, or more formally "given a particular emotion, what are the most prevalent events (or situations, contexts) that provoke it?"[1] Information about these *emotion-provoking events* is potentially useful for emotion recognition (recognizing emotion based on events mentioned in a dialogue), response generation (providing an answer to emotion-related questions), and answering social-science related questions (discovering events that affect the emotion of a particular segment of the population).

[1]This is in contrast to existing sentiment lexicons (Riloff et al., 2003; Valitutti, 2004; Esuli and Sebastiani, 2006; Velikovich et al., 2010; Mohammad and Turney, 2013), which only record the sentiment orientation of particular words (such as "meet" or "friend"), which, while useful, are less directly connected to the emotions than the events themselves.

While there is very little previous research on this subject, one previous work of note by Tokuhisa et al. (2008) focused on emotion-provoking events purely from the viewpoint of emotion recognition. They used large corpus of examples collected from the Web using manual patterns to build a k-nearest-neighbors emotion classifier for dialog systems and found that the classifier significantly outperforms baseline methods. This method provides both an inspiration and a baseline for our work, but still lacks in that it makes no attempt to measure the quality of the extracted events, aggregate similar events, or rank events by prevalence, all essential factors when attempting to use extracted events for applications other than simple emotion recognition.

In this paper, we describe work on creating prevalence-ranked dictionaries of emotion-provoking events through both manual labor and automatic information extraction. To create a manual dictionary of events, we perform a survey asking 30 participants to describe events that caused them to feel a particular emotion, and manually cleaned and aggregated the results into a ranked list. Next, we propose several methods for extracting events automatically from large data from the Web, which will allow us to increase the coverage over the smaller manually created dictionary. We start with Tokuhisa et al. (2008)'s patterns as a baseline, and examine methods for improving precision and coverage through the use of seed expansion and clustering. Finally, we discuss evaluation measures for the proposed task, and perform an evaluation of the automatically extracted emotion-provoking events. The acquired events will be provided publicly upon acceptance of the paper.

2 Manual Creation of Events

In order to create a small but clean set of gold-standard data for each emotion, we first performed

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 128–132,
Gothenburg, Sweden, April 26-30 2014.

Emotions	Words
happiness	happy, glad
sadness	sad, upset
anger	angry, irritated
fear	afraid, scared
surprise	surprised, astonished
disgust	disgusted, terrible

Table 2: Seed words for each emotion.

a survey on emotion-provoking events. We did so by asking a total of 30 subjects (a mixture of male and female from 20-40 years of age) to write down five events that provoke each of five emotions: happiness, sadness, anger, fear, and surprise. As these events created according to this survey still have a large amount of lexical variation, we manually simplify them to their core and merge together events that have similar meanings.

Finally, for each emotion we extract all the events that are shared by more than one person. It should be noted that this will not come anywhere close to covering the entirety of human emotion, but as each event is shared by at least two people in a relatively small sample, any attempt to create a comprehensive dictionary of emotion-provoking events should at least be able to cover the pairs in this collection. We show the most common three events for each emotion in Table 1.

3 Automatic Extraction of Events

We also performed experiments attempting to automatically extract and aggregate events from Web data. As a starting point, we follow Tokuhisa et al. (2008) in defining a single reliable pattern as a starting point for event extraction:

I am EMOTION that EVENT

As this pattern is a relatively reliable indicator that the event is correct, most events extracted by this pattern will actually be emotion-provoking events. For instance, this pattern will be matched with the sentence "I am *happy* that *my mother is feeling better*", in which *my mother is feeling better* certainly causes happiness.

For the EMOTION placeholder, we take into account 6 emotions - happiness, sadness, anger, fear, disgust, and surprise - argued by Ekman (1992) to be the most basic. We manually create a short list of words that can be inserted into the above pattern appropriately, as shown in Table 2.

For the EVENT placeholder, we allow any string of words, but it is necessary to choose the scope of the string that is referring to the emotion-provoking event. To this end, we use a syntactic parser and set a hard restriction that all events must be a subtree having root tag S and containing at least one noun phrase and one verb phrase.

Given these two restrictions, these patterns provide us with high quality event-emotion pairs, but the method is still lacking in two respects, lack of coverage and lack of ability to aggregate similar events. As both of these are essential to creating a high-quality and non-redundant dictionary of events, we make two simple extensions to the extraction process as follows.

3.1 Pattern Expansion

Pattern expansion, or bootstrapping algorithms are widely used in the information extraction field (Ravichandran and Hovy, 2002). In particular Espresso (Pantel and Pennacchiotti, 2006) is known as a state-of-the-art pattern expansion algorithm widely used in acquiring relationships between entities. We omit the details of the algorithm for space concerns, but note that applying the algorithm to our proposed task is relatively straightforward, and allows us to acquire additional patterns that may be matched to improve the coverage over the single seed pattern. We do, however, make two changes to the algorithm. The first is that, as we are interested in extracting events instead of entities, we impose the previously mentioned restriction of one verb phrase and one noun phrase over all events extracted by the patterns. The second is that we perform normalization of events to reduce their variability, namely removing all function words, replacing proper nouns with special symbol, and lemmatizing words.

3.2 Grouping events

The second improvement we perform is grouping the extracted events together. Grouping has a number of potential practical advantages, as noted frequently in previous work (Becker et al., 2011). The first is that by grouping similar events together, we can relieve sparsity issues to some extent by sharing statistics among the events in a single group. The second is that aggregating events together allows humans to browse the lists more efficiently by reducing the number of redundant entries. In preliminary experiments, we attempted several clustering methods and even-

Emotions	Events		
happiness	meeting friends	going on a date	getting something I want
sadness	someone dies/gets sick	someone insults me	people leave me alone
anger	someone insults me	someone breaks a promise	someone is too lazy
fear	thinking about the future	taking a test	walking/driving at night
surprise	seeing a friend unexpectedly	someone comes to visit	receiving a gift

Table 1: The top three events for each emotion.

tually settled on hierarchical agglomerative clustering and the single-linkage criterion using cosine similarity as a distance measure (Gower and Ross, 1969). Choosing the stopping criterion for agglomerative clustering is somewhat subjective, in many cases application dependent, but for the evaluation in this work, we heuristically choose the number of groups so the average number of events in each group is four, and leave a further investigation of the tuning to future work.

4 Evaluation Measures

Work on information extraction typically uses accuracy and recall of the extracted information as an evaluation measure. However, in this work, we found that it is difficult to assign a clear-cut distinction between whether an event provokes a particular emotion or not. In addition, recall is difficult to measure, as there are essentially infinitely many events. Thus, in this section, we propose two new evaluation measures to measure the precision and recall of the events that we recovered in this task.

To evaluate the precision of the events extracted by our method, we focus on the fact that an event might provoke multiple emotions, but usually these emotions can be ranked in prominence or appropriateness. This is, in a way, similar to the case of information retrieval, where there may be many search results, but some are more appropriate than others. Based on this observation, we follow the information retrieval literature (Voorhees, 1999) in adapting mean reciprocal rank (MRR) as an evaluation measure of the accuracy of our extraction. In our case, one event can have multiple emotions, so for each event that the system outputs, we ask an annotator to assign emotions in descending order of prominence or appropriateness, and assess MRR with respect to these ranked emotions. [2]

We also measure recall with respect to the manually created dictionary described in Section 2, which gives us an idea of what percent of common emotions we were able to recover. It should be noted that in order to measure recall, it is necessary to take a matching between the events output by the system and the events in the previously described list. While it would be ideal to do this automatically, this is difficult due to small lexical variations between the system output and the list. Thus, for the current work we perform manual matching between the system hypotheses and the references, and hope to examine other ways of matching in future work.

5 Experiments

In this section, we describe an experimental evaluation of the accuracy of automatic extraction of emotion-provoking events.

5.1 Experimental Setup

We use Twitter[3] as a source of data, as it is it provides a massive amount of information, and also because users tend to write about what they are doing as well as their thoughts, feelings and emotions. We use a data set that contains more than 30M English tweets posted during the course of six weeks in June and July of 2012. To remove noise, we perform a variety of preprocessing, removing emoticons and tags, normalizing using the scripts provided by Han and Baldwin (2011), and Han et al. (2012). CoreNLP[4] was used to get the information about part-of-speech, syntactic parses, and lemmas.

We prepared four systems for comparison. As a baseline, we use a method that only uses the original seed pattern mentioned in Section 3 to acquire emotion-provoking events. We also evaluate expansions to this method with clustering, with pattern expansion, and with both.

We set a 10 iteration limit on the Espresso algorithm and after each iteration, we add the 20

[2] In the current work we did not allow annotators to assign "ties" between the emotions, but this could be accommodated in the MRR framework.

[3] http://www.twitter.com

[4] http://nlp.stanford.edu/software/corenlp.shtml

Methods	MRR	Recall
Seed	46.3 ($\pm$5.0)	4.6 ($\pm$0.5)
Seed + clust	57.2 ($\pm$7.9)	8.5 ($\pm$0.9)
Espresso	49.4 ($\pm$2.8)	8.0 ($\pm$0.5)
Espresso + clust	**71.7** ($\pm$2.9)	**15.4** ($\pm$0.8)

Table 3: MRR and recall of extracted data (with standard deviation for 3 annotators).

Emotions	MRR	Recall
happiness	93.9	23.1
sadness	76.9	10.0
anger	76.5	14.0
fear	48.3	24.3
surprise	59.6	0.0

Table 4: Average MRR and recall by emotion for the Espresso + clustering method.

most reliable patterns to the pattern set, and increase the seed set by one third of its size. These values were set according to a manual inspection of the results for several settings, before any evaluation was performed.

We examine the utility of each method according to the evaluation measures proposed in Section 4 over five emotions, happiness, sadness, anger, fear, and surprise.[5] To measure MRR and recall, we used the 20 most frequent events or groups extracted by each method for these five emotions, and thus all measures can be interpreted as MRR@20 and recall@20. As manual annotation is required to calculate both measures, we acquired results for 3 annotators and report the average and standard deviation.

5.2 Experimental Results

The results are found in Table 3. From these results we can see that clustering the events causes a significant gain on both MRR and recall, regardless of whether we use Espresso or not. Looking at the results for Espresso, we see that it allows for small boost in recall when used on its own, due to the fact that the additional patterns help recover more instances of each event, making the estimate of frequency counts more robust. However, Espresso is more effective when used in combination with clustering, showing that both methods are capturing different varieties of information, both of which are useful for the task.

In the end, the combination of pattern expansion and clustering achieves an MRR of 71.7% and recall of 15.4%. While the MRR could be deemed satisfactory, the recall is still relatively low. One reason for this is that due to the labor-intensive manual evaluation, it is not realistic to check many more than the top 20 extracted events for each emotion, making automatic evaluation metrics the top on the agenda for future work.

[5]We exclude disgust, as the seed only matched 26 times over entire corpus, not enough for a reasonable evaluation.

However, even without considering this, we found that the events extracted from Twitter were somewhat biased towards common, everyday events, or events regarding love and dating. On the other hand, our annotators produced a wide variety of events including both everyday events, and events that do not happen every day, but leave a particularly strong impression when encountered. This can be seen particularly in the accuracy and recall results by emotion for the best system shown in Table 4. We can see that for some emotions we achieved recall approaching 25%, but for surprise we didn't manage to extract any of the emotions created by the annotators at all, instead extracting more mundane events such as "surprised I'm not fat yet" or "surprised my mom hasn't called me yet." Covering the rare, but important events is an interesting challenge for expansions to this work.

6 Conclusion and Future Work

In this paper we described our work in creating a dictionary of emotion-provoking events, and demonstrated results for four varieties of automatic information extraction to expand this dictionary. As this is the first attempt at acquiring dictionaries of emotion-provoking events, there are still many future directions that deserve further investigation. As mentioned in the experimental discussion, automatic matching for the evaluation of event extraction, and ways to improve recall over rarer but more impressive events are necessary. There are also many improvements that could be made to the extraction algorithm itself, including more sophisticated clustering and pattern expansion algorithms. Finally, it would be quite interesting to use the proposed method as a tool for psychological inquiry, including into the differences between events that are extracted from Twitter and other media, or the differences between different demographics.

References

Hila Becker, Mor Naaman, and Luis Gravano. 2011. Beyond trending topics: Real-world event identification on Twitter. In *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media (ICWSM11)*.

Paul Ekman. 1992. An argument for basic emotions. *Cognition & Emotion*, 6(3-4):169–200.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation*, pages 417–422.

John C Gower and GJS Ross. 1969. Minimum spanning trees and single linkage cluster analysis. *Applied statistics*, pages 54–64.

Bo Han and Timothy Baldwin. 2011. Lexical normalisation of short text messages: makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 368–378.

Bo Han, Paul Cook, and Timothy Baldwin. 2012. Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 421–432, Jeju Island, Korea, July. Association for Computational Linguistics.

Saif M Mohammad and Peter D Turney. 2013. Crowdsourcing a word–emotion association lexicon. *Computational Intelligence*, 29(3):436–465.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 113–120.

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 41–47.

Ellen Riloff, Janyce Wiebe, and Theresa Wilson. 2003. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 25–32. Association for Computational Linguistics.

Ryoko Tokuhisa, Kentaro Inui, and Yuji Matsumoto. 2008. Emotion classification using massive examples extracted from the web. In *Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 881–888.

Ro Valitutti. 2004. Wordnet-affect: an affective extension of wordnet. In *In Proceedings of the 4th International Conference on Language Resources and Evaluation*, pages 1083–1086.

Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 777–785.

Ellen M Voorhees. 1999. The trec-8 question answering track report. In *Proceedings of TREC*, volume 99, pages 77–82.

Chinese Temporal Tagging with HeidelTime

Hui Li and **Jannik Strötgen** and **Julian Zell** and **Michael Gertz**
Institute of Computer Science, Heidelberg University
Im Neuenheimer Feld 348, 69120 Heidelberg, Germany
{hui.li,stroetgen,zell,gertz}@informatik.uni-heidelberg.de

Abstract

Temporal information is important for many NLP tasks, and there has been extensive research on temporal tagging with a particular focus on English texts. Recently, other languages have also been addressed, e.g., HeidelTime was extended to process eight languages. Chinese temporal tagging has achieved less attention, and no Chinese temporal tagger is publicly available. In this paper, we address the full task of Chinese temporal tagging (extraction and normalization) by developing Chinese HeidelTime resources. Our evaluation on a publicly available corpus – which we also partially re-annotated due to its rather low quality – demonstrates the effectiveness of our approach, and we outperform a recent approach to normalize temporal expressions. The Chinese HeidelTime resource as well as the corrected corpus are made publicly available.

1 Introduction

Temporal information plays a crucial role in many documents, and temporal tagging, i.e., the extraction of temporal expressions and their normalization to some standard format, is crucial for several NLP tasks. So far, research on temporal information extraction mostly focused on western languages, especially English. In contrast, eastern languages, e.g., Chinese, are less explored. Nevertheless, there is research on Chinese temporal tagging. While some works addressed either the extraction or the normalization subtask, a few full temporal taggers exist, e.g., CTEMP (Wu et al., 2005b) and CTAT (Jing et al., 2008), but none of them is publicly available.

In contrast, some temporal taggers were recently made available, e.g., DANTE (Mazur and Dale, 2009), TipSem (Llorens et al., 2010), and HeidelTime (Strötgen and Gertz, 2013). Furthermore, Strötgen et al. (2013) showed that HeidelTime can be extended to further languages by developing language-specific resources without modifying the source code. Thus, when developing temporal tagging capabilities for an additional language, one is faced with the question of whether to develop a new temporal tagger or to extend an existing one. We decided to extend HeidelTime for Chinese for the following reasons: (i) HeidelTime was the best temporal tagger in the TempEval-2 (English) and TempEval-3 (English and Spanish) competitions (Verhagen et al., 2010; UzZaman et al., 2013), (ii) it already supports eight languages, and (iii) it is the only multilingual temporal tagger for cross-domain temporal tagging, e.g., news- and narrative-style documents can be processed with high quality.

2 Related Work

For Chinese temporal tagging, machine learning and rule-based approaches have been employed. Wu et al. (2005a) and Wu (2010) report that machine learning techniques do not achieve as good results as rule-based approaches when processing Chinese. Thus, it is reasonable to extend a rule-based system such as HeidelTime to Chinese.

In general, temporal tagging approaches perform the extraction, the normalization, or both, and create TIDES TIMEX2 (Ferro et al., 2005) or TimeML's TIMEX3 (Pustejovsky et al., 2005) annotations. For development and evaluation, there are two Chinese temporally annotated corpora, the ACE 2005 training corpus and TempEval-2 (c.f. Section 3). Table 1 lists approaches to Chinese temporal tagging with some further information. The most recent work is the learning-based language-independent discriminative parsing approach for normalizing temporal expressions by Angeli and Uszkoreit (2013).

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 133–137,
Gothenburg, Sweden, April 26-30 2014.

approach	tasks	method	standard	evaluation details	system available
Angeli and Uszkoreit (2013)	N	ML	TIMEX3	TempEval-2 (N)	no
Wu (2010)#	E	rules	TIMEX2	ACE 2007 (E)	no
Wen (2010)#	N	rules	TIMEX2	own corpus (N)	no
He (2009)#	E	ML+rules	TIMEX2	ACE 2005 (E)	no
Pan (2008)#	E	ML+rules	TIMEX2	ACE 2005 (E)	no
Jing et al. (2008)# – CTAT	E+N	ML+rules	TIMEX2	own corpus (E+N)	no
Wu et al. (2005b) – CTEMP	E+N	rules	TIMEX2	TERN 2004 (E), own corpus (E+N)	no
Hacioglu et al. (2005) – ATEL	E	ML+rules	TIMEX2	TERN 2004 (E)	no

Table 1: Information on related work addressing Chinese temporal tagging (#available in Chinese only).

There are also (semi-)automatic approaches to port a temporal tagger from one language to another. For instance, TERSEO (Negri et al., 2006; Saquete et al., 2006) has been extended from Spanish to English and Italian by automatic rule-translation and automatically developed parallel corpora. However, the normalization quality of this approach was rather low compared to a rule-based tagger manually developed for the specific language (Negri, 2007). This finding encouraged us to manually create Chinese HeidelTime resources instead of trying automatic methods.

3 The TempEval-2 Chinese Corpus

There are two Chinese temporally annotated corpora available: While the Chinese part of the ACE 2005 multilingual training corpus (Walker et al., 2006) has been used by some approaches (c.f. Table 1), it only contains TIMEX2 extent annotations. In contrast, the TempEval-2 Chinese data sets (Verhagen et al., 2010) contain TIMEX3 annotations with extent and normalization information. However, no TempEval-2 participants addressed Chinese and only Angeli and Uszkoreit (2013) report evaluation results on this corpus. Since HeidelTime is TIMEX3-compliant, and we address the extraction and normalization subtasks, we use the TempEval-2 corpus in our work.

3.1 Annotation Standard TimeML

For temporal expressions, TimeML (Pustejovsky et al., 2005) contains TIMEX3 tags with several attributes. The two most important ones – also annotated in the TempEval-2 data – are *type* and *value*. Type specifies if an expression is a date, time, duration, or set (set of times), and value contains the normalized meaning in standard format.

3.2 Original TempEval-2 Corpus

The Chinese training and test sets consist of 44 and 15 documents with 746 and 190 temporal expressions, respectively. However, several expressions have no normalized value information (85 in the training and 47 in the test set), others no type.[1]

This issue was also reported by Angeli and Uszkoreit (2013). Thus, they report evaluation results on two versions of the data sets, the original version and a cleaned version, in which all expressions without value information were removed.

3.3 Re-annotation of the TempEval-2 Corpus

Due to the significant amount of temporal expressions with undefined value attributes, we decided to manually assign normalized values to these expressions instead of excluding them. During this process, we recognized that the corpus contained several more errors, e.g., some expressions were annotated as dates although they refer to durations. Thus, instead of only substituting undefined values, we checked all annotations in the two data sets and corrected errors. For this, one Chinese native and two further TimeML experts discussed all modified annotations. Although there were several difficult expressions and not all normalizations were straightforward, we significantly improved the annotation quality. After our modification, the improved training and test sets contain 765 and 193 temporal expressions with value information, respectively. In Table 2, statistics about the three versions of the data sets are provided.

4 Chinese HeidelTime Resources

HeidelTime is a cross-domain, multilingual temporal tagger that strictly separates the source code and language-dependent resources (Strötgen and Gertz, 2013). While the implementation takes care of domain-dependent normalization issues, language-dependent resources contain pattern, normalization, and rule files. We had to develop such Chinese resources to perform Chinese temporal tagging with HeidelTime.

[1]Note that the TempEval-2 corpus developers stated that the annotations of the non-English documents are rather experimental (Verhagen, 2011).

corpus	docs	temp. expr.	date / time / duration / set	undef value
training set				
original	44	746	623 / 10 / 113 / 0	85
AU13-clean	44	661	555 / 10 / 96 / 0	0
improved	44	765	628 / 10 / 125 / 2	0
test set				
original	15	190	160 / 0 / 27 / 3	47
AU13-clean	15	143	128 / 0 / 15 / 0	0
improved	15	193	166 / 0 / 23 / 4	0

Table 2: Statistics on the three versions of the Chinese TempEval-2 data sets.

4.1 Chinese Linguistic Preprocessing

As input, HeidelTime requires sentence, token, and part-of-speech information. For most of the supported languages, HeidelTime uses a UIMA wrapper of the TreeTagger (Schmid, 1994). Since there is also a Chinese model for the TreeTagger available, we rely on the TreeTagger for Chinese linguistic preprocessing.[2]

4.2 Resource Development Process

To develop Chinese HeidelTime resources, we followed the strategy applied by Strötgen et al. (2013) for Spanish: Using HeidelTime's English resources as starting point, we translated the pattern files, the normalization files, and the rules for extracting and normalizing temporal expressions. More details on these steps are provided next.

Pattern & Normalization Resources. English patterns in the pattern files, which also exist in Chinese in a similar form, were directly translated. For instance, there are Chinese expressions for names of months and weekdays. Patterns existing in English but not used in Chinese were removed, e.g., there are no abbreviations of month names in Chinese. In contrast, for other patterns frequently used in Chinese, additional pattern files were created. Examples are Chinese numerals.

Based on the pattern files, we built the normalization files. Here, the normalized values of the patterns are stored. An example of the Chinese resources is as follows: The three patterns "星期二", "礼拜二", and "周二" can all be translated as Tuesday and are thus part of the Weekday pattern resource. Since weekdays are internally handled by HeidelTime with their English names, the normalization file for Chinese weekdays contains "星期二,Tuesday" "礼拜二,Tuesday" and "周二,Tuesday".

[2] http://corpus.leeds.ac.uk/tools/zh/.

Chinese Rule Development. HeidelTime's rules contain three components, a name, an extraction and a normalization part. The extraction mainly makes use of regular expressions and the pattern resources, and in the normalization part, the matched expressions are normalized using the normalization resources.[3] To develop the rules, we again followed Strötgen et al. (2013) and applied the following strategy:

(i) A few simple Chinese rules were created based on the English rules. (ii) We reviewed extracted temporal expressions in the training set and improved the extraction and normalization parts of the rules. (iii) We checked the training texts for undetected expressions and created rules to match them. In parallel, we adapted the Chinese pattern and normalization resources when necessary. (iv) We translated more complex English rules to also cover valid expressions not occurring in the Chinese training documents. (v) Steps (ii) to (iv) were iteratively performed until the results on the training set could not be improved further.

4.3 Chinese Challenges

Chinese is an isolating language without inflection and depends on word order and function words to represent grammatical relations. Although we only consider modern Mandarin as it is the most widely used variety of Chinese in contemporary texts, many challenges occurred during the resource development process. Some examples are:

Polysemous words: Many Chinese words have more than one meaning, e.g., dynasty names such as "唐" (Tang) or "宋" (Song) can refer to a certain time period, but also appear as family names.

Further ambiguities: There are many ambiguous expressions in Chinese, e.g., the temporal expression "五日前" has two meanings: "before the 5th day of a certain month" and also "5 days ago" – depending on the context.

Calendars: There are various calendars in Chinese culture and thus also in Chinese texts, such as the lunar calendar and the 24 solar terms, which are different from the Gregorian calendar and thus very difficult to normalize. Besides, Taiwan has a different calendar, which numbers the year from the founding year of the Republic of China (1911).

[3] For more details about HeidelTime's system architecture and rule syntax, we refer to Strötgen and Gertz (2013).

training set	P	R	F	value	type
original	96.1	92.7	94.4	80	93
AU13-clean	80.7	95.1	87.3	91	95
improved	97.6	94.4	96.0	92	95
test set	P	R	F	value	type
original	93.4	82.0	87.3	70	93
AU13-clean	63.5	88.0	73.8	89	96
improved	95.5	83.8	89.3	87	96

Table 3: Evaluation results for extraction and normalization (TempEval-2 training and test sets).

	original		AU13-clean		# correct
training set	value	type	value	type	value
AU13	65%	95%	73%	97%	484[5]
HeidelTime	80%	93%	91%	95%	574
	original		AU13-clean		# correct
test set	value	type	value	type	value
AU13	48%	87%	60%	97%	86[5]
HeidelTime	70%	93%	89%	96%	121

Table 4: Normalization only – comparison to AU13 (Angeli and Uszkoreit, 2013).

5 Evaluation

In this section, we present evaluation results of our newly developed Chinese HeidelTime resources. In addition, we compare our results for the normalization sub-task to Angeli and Uszkoreit (2013).

5.1 Evaluation Setup

Corpus: We use three versions of the TempEval-2 training and test sets: (i) the original versions, (ii) the improved versions described in Section 3.3, and (iii) the cleaned versions also used by Angeli and Uszkoreit (2013) in which temporal expressions without value information are removed.

Setting: Since the TempEval-2 data already contains sentence and token information, we only had to perform part-of-speech tagging as linguistic preprocessing step. For this, we used the TreeTagger (Schmid, 1994) with its Chinese model.

Measures: We use the official TempEval-2 evaluation script. For the extraction, precision, recall, and f-score are calculated on the token-level. For the normalization, accuracy for the attributes *type* and *value* are calculated on the expression-level. Note that the use of accuracy makes it difficult to compare systems having a different recall in the extraction, as will be explained below.

5.2 Evaluation Results

Table 3 (top) shows the evaluation results on the training set. Extraction and normalization quality are high, and value accuracies of over 90% on the cleaned and improved versions are promising.[4]

The results on the test sets (Table 3, bottom) are lower than on the training sets. However, value accuracies of almost 90% with a recall of more than 80% are valuable and comparable to state-of-the-art systems in other languages. A first error analysis revealed that while the training documents are written in modern Mandarin, some test documents contain Taiwan-specific expressions (c.f. Section 4.3) not covered by our rules yet.

Finally, we compare the normalization quality of our approach to the multilingual parsing approach of Angeli and Uszkoreit (2013). However, their approach performs only the normalization subtask assuming that the extents of temporal expressions are provided. For this, they used gold extents for evaluation. HeidelTime only normalizes those expressions that it knows how to extract. Thus, we run HeidelTime performing the extraction and the normalization. However, since the accuracy measure used by the TempEval-2 script calculates the ratio of correctly normalized expressions to all extracted expressions and not to all expressions in the gold standard, we additionally present the raw numbers of correctly normalized expressions for the two systems. Table 4 shows the comparison between our approach and the one by Angeli and Uszkoreit (2013). We outperform their approach not only with respect to the accuracy but also with respect to the numbers of correctly normalized expressions (574 vs. 484[5] and 121 vs. 86[5] on the training and test sets, respectively) – despite the fact that we perform the full task of temporal tagging and not only the normalization.

6 Conclusions & Ongoing Work

In this paper, we addressed Chinese temporal tagging by developing Chinese HeidelTime resources. These make HeidelTime the first publicly available Chinese temporal tagger. Our evaluation showed the high quality of the new HeidelTime resources, and we outperform a recent normalization approach. Furthermore, the re-annotated Chinese TempEval-2 data sets will also be made available.

Currently, we are performing a detailed error analysis and hope to gain insights to further improve HeidelTime's Chinese resources.

[4]Note that the lower value accuracy on the original set is due to expressions without value information in the gold standard, and that the low extraction precision in the clean version is due to some of those expressions being (correctly) extracted by the system but removed from the gold standard.

[5]Number of correct value normalizations calculated based on value accuracy and number of expressions in the data sets.

References

Gabor Angeli and Jakob Uszkoreit. 2013. Language-Independent Discriminative Parsing of Temporal Expressions. In *P*roceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013), pages 83–92.

Lisa Ferro, Laurie Gerber, Inderjeet Mani, Beth Sundheim, and George Wilson. 2005. TIDES 2005 Standard for the Annotation of Temporal Expressions. Technical report, The MITRE Corporation.

Kadri Hacioglu, Ying Chen, and Benjamin Douglas. 2005. Automatic Time Expression Labeling for English and Chinese Text. In *P*roceedings of the 6th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2005), pages 548–559.

Ruifang He. 2009. *Research on Relevant Techniques of Temporal Multi-document Summarization*. Ph.D. thesis, Harbin Institute of Technology.

Lin Jing, Cao Defang, and Yuan Chunfa. 2008. Automatic TIMEX2 Tagging of Chinese Temporal Information. *J*ournal of Tsinghua University, 48(1):117–120.

Hector Llorens, Estela Saquete, and Borja Navarro. 2010. TIPSem (English and Spanish): Evaluating CRFs and Semantic Roles in TempEval-2. In *P*roceedings of the 5th International Workshop on Semantic Evaluation (SemEval 2010), pages 284–291.

Pawel Mazur and Robert Dale. 2009. The DANTE Temporal Expression Tagger. In *P*roceedings of the 3rd Language and Technology Conference (LTC 2009), pages 245–257.

Matteo Negri, Estela Saquete, Patricio Martínez-Barco, and Rafael Muñoz. 2006. Evaluating Knowledge-based Approaches to the Multilingual Extension of a Temporal Expression Normalizer. In *P*roceedings of the Workshop on Annotating and Reasoning about Time and Events (ARTE 2006), pages 30–37.

Matteo Negri. 2007. Dealing with Italian Temporal Expressions: The ITA-CHRONOS System. In *P*roceedings of EVALITA 2007, pages 58–59.

Yuequn Pan. 2008. Research on Temporal Information Recognition and Normalization. Master's thesis, Harbin Institute of Technology.

James Pustejovsky, Robert Knippen, Jessica Littman, and Roser Sauri. 2005. Temporal and Event Information in Natural Language Text. *L*anguage Resources and Evaluation, 39(2-3):123–164.

E. Saquete, P. Martínez-Barco, R. Muñoz, M. Negri, M. Speranza, and R. Sprugnoli. 2006. Multilingual extension of a temporal expression normalizer using annotated corpora. In *P*roceedings of the EACL 2006 Workshop on Cross-Language Knowledge Induction, pages 1–8.

Helmut Schmid. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. In *P*roceedings of the International Conference on New Methods in Language Processing.

Jannik Strötgen and Michael Gertz. 2013. Multilingual and Cross-domain Temporal Tagging. *L*anguage Resources and Evaluation, 47(2):269–298.

Jannik Strötgen, Julian Zell, and Michael Gertz. 2013. HeidelTime: Tuning English and Developing Spanish Resources for TempEval-3. In *P*roceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013), pages 15–19.

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. SemEval-2013 Task 1: TempEval-3: Evaluating Time Expressions, Events, and Temporal Relations. In *P*roceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013), pages 1–9.

Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. SemEval-2010 Task 13: TempEval-2. In *P*roceedings of the 5th International Workshop on Semantic Evaluation (SemEval 2010), pages 57–62.

Marc Verhagen. 2011. TempEval2 Data – Release Notes. Technical report, Brandeis University.

Christopher Walker, Stephanie Strassel, Julie Medero, and Kazuaki Maeda. 2006. ACE 2005 Multilingual Training Corpus. Linguistic Data Consortium, Philadelphia.

Yanxia Wen. 2010. Research on Time Standardization in Chinese. Master's thesis, Shanxi University.

Mingli Wu, Wenjie Li, Qing Chen, and Qin Lu. 2005a. Normalizing Chinese Temporal Expressions with Multi-label Classification. In *P*roceedings of the IEEE International Conference on Natural Language Processing and Knowledge Engineering (NLP-KE 2005), pages 318–323.

Mingli Wu, Wenjie Li, Qin Lu, and Baoli Li. 2005b. CTEMP: A Chinese Temporal Parser for Extracting and Normalizing Temporal Information. In *P*roceedings of the 2nd International Joint Conference on Natural Language Processing (IJCNLP 2005), pages 694–706.

Tong Wu. 2010. Research on Chinese Time Expression Recognition. Master's thesis, Fudan University.

A Probabilistic Approach to Persian Ezafe Recognition

Habibollah Asghari
Department of ECE,
University of Tehran,
Tehran, Iran
habib.asghari@ut.ac.ir

Jalal Maleki
Department of CIS,
Linköpings Universitet
SE-581 83 Linköping, Sweden
jalal.maleki@liu.se

Heshaam Faili
Department of ECE,
University of Tehran,
Tehran, Iran
hfaili@ut.ac.ir

Abstract

In this paper, we investigate the problem of Ezafe recognition in Persian language. Ezafe is an unstressed vowel that is usually not written, but is intelligently recognized and pronounced by human. Ezafe marker can be placed into noun phrases, adjective phrases and some prepositional phrases linking the head and modifiers. Ezafe recognition in Persian is indeed a homograph disambiguation problem, which is a useful task for some language applications in Persian like TTS. In this paper, Part of Speech tags augmented by Ezafe marker (POSE) have been used to train a probabilistic model for Ezafe recognition. In order to build this model, a ten million word tagged corpus was used for training the system. For building the probabilistic model, three different approaches were used; Maximum Entropy POSE tagger, Conditional Random Fields (CRF) POSE tagger and also a statistical machine translation approach based on parallel corpus. It is shown that comparing to previous works, the use of CRF POSE tagger can achieve outstanding results.

1 Introduction

In Persian language, Ezafe is an unstressed short vowel /-e/ (or /-ye/ after vowels) which is used to link two words in some contexts. Although Ezafe is an important part of the Persian phonology and morphology, it does not have a specific character representation, and so is not usually written. However, it is pronounced as a short vowel /e/. Sometimes, for disambiguation purposes it is preferred to explicitly mark its presence by a written symbol (the diacritic Kasre) after some words in order to facilitate the correct pronunciation.

The most important application of Ezafe recognition is a text to phoneme tool for Text To Speech (TTS) Systems. Other application of Ezafe recognition is identifying the dependency of a word in a Noun Phrase. (Oskouipour, 2011, Mavvaji and Eslami, 2012)

In this research, we would like to investigate various approaches to correctly recognize genitive cases in Persian language. Shortly, the contributions of this paper are as follow:

- Modeling the Ezafe recognition task as a sequence labeling system.
- Using HMM and CRF as sequence labelers.
- Modeling the Ezafe recognition task as a monotone translation problem which can be tackled by phrase based SMT approach.
- Using a big amount of test and gold data, so the results are considerably reliable.
- To enhance the results of the system, five Persian-specific features which discriminate the results in high-precision low-recall fashion, have been proposed.
- The recognition rate has achieved outstanding results in comparison to the previous works.

This task is closely related to the task of determining short vowels in Arabic language. So, although the aim of this paper is to recognize Ezafe in Persian language, but all the methods investigated here is applicable to determine short vowels in Arabic language.

In the next section a clear definition of the problem is presented and the characteristics of Persian language are introduced. In Section 3 we will give a precise definition of Ezafe. Section 4 provides an overview of previous works on Ezafe recognition. Our approach will be described in Section 5 followed by two sections including corpus selection process and implementation of proposed method. Conclusion and recommendations for future works will be presented in the last section.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 138–142,
Gothenburg, Sweden, April 26-30 2014.

2 An Overview of Persian Language

Persian Language belongs to Arabic script-based languages. This category of languages includes Kurdish, Urdu, Arabic, Pashtu and Persian. They all have common scripting, and somehow similar writing system.

In Arabic script-based languages, the most common features are absence of capitalization, right to left direction, lack of clear word boundaries, complex word structure, encoding issues in computer environment, and a high degree of ambiguity due to non-representation of short vowels in writing (Farghaly, 2004). Note that Ezafe recognition and homograph disambiguation problem mostly deals with the last mentioned feature.

One of the problems in Persian language processing is long-distance dependencies. This phenomenon complicates Ezafe recognition task even for humans (Ghomeshi, 1996). Another problem is how to determine phrase/word boundaries. In Persian language, affixes can be written in three formats; completely separated by a space delimiter, separated by half-space[1], or can be attached to its main word. So, determining word and phrase boundaries are somehow a complicated task in Persian.The third challenge arises by pronoun drop due to the morphology of Persian language.

3 Ezafe Definition

Historically, Persian Ezafe had a demonstrative morpheme in old Iran (Estaji and Jahangiri, 2006). It was related to a demonstrative /hya/, which links the head noun to adjectival modifiers, to the possessor NP (Samvelian, P., 2007). In evolution of Persian language, /hya/ became /–i/ in Middle Persian and progressively lost its demonstrative value to end as a simple linker. In recognizing Ezafe, we should consider all morphological, syntactic, semantic and discourse views (Parsafar, 2010). It should be noted that Ezafe can be iterated within the NP, occurring as many times as there are modifiers.

4 Previous Works

As a first attempt to recognize Ezafe in Persian text, Bijankhan (Bijankhan, 2005) used a pattern matching algorithm for Ezafe recognition. He has used POS tags and also semantic labels (such as place, time, ordinal numbers ...) to obtain a statistical view of Ezafe markers. He manually derived 80 most frequent patterns such as Noun-Noun and Noun-Adjective etc. The most frequent combinations were extracted based on a 10 million-wordscorpus.

In a research accomplished by (Isapour, et al., 2007), the researchers rely on the fact that Ezafe can relate between head and its modifiers so as to help to build NPs. So by parsing sentences and finding Phrase borders, the location of Ezafe in the sentence can be found. In this work, the sentences were analyzed using a Probabilistic Context Free Grammar (PCFG) to derive phrase borders. Then based on the extracted parse tree, the head and modifiers in each phrase can be determined. In the last phase, a rule based approach was also applied to increase the accuracy in Ezafe marker labeling. For training the algorithm, 1000 sentences were selected and a parse tree was built for each of them. Because of the limited number of parsed sentences for training, the results cannot be extended for general applications.

There were also other attempts to effectively recognize Ezafe marker in Persian text, such as (Zahedi, 1998) based on fuzzy sets. Also, (Oskouipour, 2011) developed a system based on Hidden Markove Model to correctly identify Ezafe markers. (Mavvaji and Eslami, 2012) had another attempt by syntactic analysis. There are also some implementations using neural networks (Razi and Eshqi, 2012). Some of the results can be seen in Table 4.

5 Our Approach

In this paper we have investigated two types of POS taggers, and also a MT-based approach. In the following section, these approaches will be explained and the results will be compared to previous work.

A. Ezafe recognition as a POS tagging problem

Part Of Speech tagging is an effective way for automatically assigning grammatical tags to words in a text. In Persian, POS tagging can be applied as a homograph disambiguation problem for correct pronunciation of words in a sentence (Yarowsky, 1996). There are powerful POS tagger algorithms such as statistical, rule based, transformation based and memory based learning methods. In this research we have used two schemes of statistical POS tagging for Ezafe recognition. The first one is a Maximum Entropy tagger that has been investigated by (Toutanova and Manning. 2000) and (Toutanova, et al.

[1]A Non-Joint Zero Width (NJZW) letter

2003). In order to implement this approach, we have used Stanford toolkit as a MaxEnt tagger. The second approach is based on Conditional Random Fields (CRF) model, that was first introduced by (Lafferty, et al., 2001) and then (Sha and Pereira. 2003).

B. Ezafe recognition as a translation problem

We can consider the Ezafe recognition problem as a monotone translation problem. In other words, it can be considered as a noisy channel problem. The original training text without the Ezafe marker can be used as source language, and the tagged text can be used as destination language. So, we can apply these parallel corpora as inputs to a phrase-based Statistical Machine Translation (SMT) system.

In the experiments, we have used monotone SMT with distortion limit equal to zero. For implementing SMT, we have used Moses toolkit. It should be mentioned that in the case of Ezafe recognition, we can use a SMT system without re-ordering. By using phrase-based SMT, the local dependencies between the neighboring words are handled by the phrase table. Also some of the dependencies between different phrases can be tackled by the language model.

6 Data Preparation

In this work, we have used Bijankhan corpus (Bijankhan, 2004, Amiri, et al, 2007). The content of this corpus is gathered from daily news and common texts, covering 4300 different subjects. It contains about 10 million tagged words in about 370000 sentences. The words in the corpus have been tagged by 550 tags based on a hierarchical order, with more fine-grained POS tags like 'noun-plural-subj'. About 23% of words in the corpus are tagged with Ezafe. We have used an extended version of POS tags, named POSE (Part of Speech tags + Ezafe tag) that can be constructed by adding Ezafe markers to original first level tags. Table 1 shows the statistics of POSE tags.

POSE	Frequency	% in Ezafe markers	% in all corpus
N-e	*1817472*	*81.87*	*18.39*
ADJ-e	*223003*	*10.05*	*2.26*
P-e	*111127*	*5.01*	*1.125*
NUM-e	*27860*	*1.26*	*0.28*
others	*40477*	*1.81*	*0.41*
Total	*2219939*	*100 %*	*22.46*

Table 1 - Ezafe Statistics in Bijankhan Corpus

7 Performance Metrics

The ordinary measures that can be used based on confusion matrix are Precision, Recall and F1 measure. Another measure that can be used in this binary classification problem is Mathews Correlation Coefficient (MCC). This measure indicates the quality of the classifier for binary class problems especially when two classes are of very different sizes.We have also considered two other measures; true positive rate as Ezafe presence accuracy, and false positive rate as Ezafe absence accuracy. The total average can be calculated using a weighted average of the two last mentioned measures.

8 Experiments and Results

As mentioned, the system was trained on Bijankhan corpus. Only the first level of POS tags was used for the training phase, except for the words with Ezafe, that the POS plus Ezafe marker was chosen. The more fine-grained POS tags were removed to achieve more accuracy.

We used a ten-fold cross-validation scheme. For calculating the total accuracy, Ezafe presence accuracy and Ezafe absence accuracy should be weighted by 16.8% (ratio of words with Ezafe marker in test corpus) and 83.2% (ratio of words without Ezafe marker) respectively.

A. Evaluating fine-grained tags

The first experiment was done in order to test the ability of other fine grained POS tags in Ezafe recognition. In this test that was done on 30% of the corpus, all of the fine grained POS tags of the words plus Ezafe marker were used to train a Maximum Entropy POSE tagger. As shown in Table 2, the accuracy of the system decreased when we used complete features hierarchy. So, in consequent experiments, we used only first level tag features.

Conditions	**Performance measures (Run on 30% of corpus)**				
	Precision	***Recall***	***F-measure***	***MCC***	***Accuracy***
MaxEnt+ POSE	*87.95*	*93.14*	*0.91*	*0.89*	*96.71*
MaxEnt+ POSE+ fine grained tags	*89.56*	*88.69*	*0.89*	*0.87*	*96.37*

Table 2: Experiment Based on Full Tag Hierarchy

B. Evaluating MaxEnt tagger

In the next experiment, we used a MaxEnt tagger applied on whole corpus. With first level hierarchy of POSE tags, a total accuracy of 97.21% was resulted. As shown in the Table 3, while we have a good recall rate, the precision

reached a fair value. Both F-measure and MCC have values greater than 0.9.

The effect of eliminating titles which are incomplete sentences was also experimented. Table 3 shows that eliminating the titlesdoes not achieve a good improvement in accuracy.

C. *Using Persian-specific features*

Augmenting the system with some Persian-specific features to decrease FP and FN can significantly increase the total accuracy. As shown in Table 3, by using five features, the accuracy can be increased by more than 0.6%. The features are as follow:

- Punctuations cannot take Ezafe. By this simple feature, these FP errors will be removed.
- Noun words which are followed by adjectives and adverbs should take Ezafe marker.
- Adjectives which are followed by nouns and adverbs should take Ezafe marker.
- Adverbs which are followed by nouns and adverbs should take Ezafe marker.
- Nouns, adverbs and adjectives which are followed by verbs do not take Ezafe.

Conditions	**Performance measures (%)**				
	Precision	***Recall***	***F-measure***	***MCC***	***Accuracy***
MAXent+POSE	*89.44*	*94.48*	*0.919*	*0.903*	*97.21*
MAXent+POSE without title	*89.53*	*94.47*	*0.919*	*0.903*	*97.23*
Maxent+POSE+ Persian Specific Features	*91.37*	*95.92*	*0.936*	*0.923*	*97.80*

Table 3 - Results of Experiments on complete corpus Size

Note that the false positive rate of the above mentioned experiment is about twice of the false negative rate. So, we tried to extract more features based on investigating words in FP table and confusion matrix.

D. *Evaluating CRF Tagger*

The next experiment was based on CRF tagger. In order to compare the results with MaxEnt tagger, the experiment was performed on whole corpus using 10-fold cross validation method.

In this experiment, we used a CRF tagger and applied a window on the text to see the effect of neighboring words as input features in Ezafe recognition. As shown in Figure 1, the accuracy of system varies by changing the size of the window from 1 to 9. The graph shows that the experiments with a CRF tagger can achieve its best accuracy with window of size 5. Better performance was achieved by augmenting the CRF model with the five mentioned Persian-specific features.

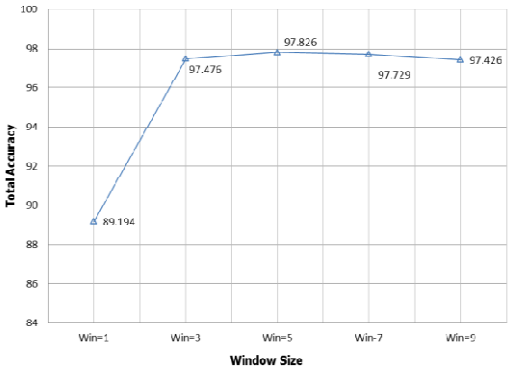


Fig. 1. Ezafe Recognition Accuracy vs. Window Size

Table 4 shows the results comparing with previous works in this regard. As shown in the table, the accuracy of CRF model augmented by mentioned featuresets can achieve best results comparing to other approaches.

Conditions	Ezafe presence accuracy	Ezafe Presence Error	Ezafe Absence Accuracy	Ezafe Absence Error	Total Accuracy
Rule based and syntactic (Oskouipour, 2011)	*10.37*	*89.63*	*83.20*	*16.80*	*70.06*
PCFG with 1000 sentences (Isapour, 2007)	*86.74*	*13.26*	*95.62*	*4.38*	*93.29*
Pattern based method patterns with freq>1% (Bijankhan, 2005)	*79.69*	*20.31*	*92.95*	*7.05*	*89.86*
HMM with 3gram (Oskouipour, 2011)	*78.55*	*21.45*	*95.31*	*4.68*	*91.69*
SMT based approach	*75.96*	*24.05*	*89.99*	*10.01*	*88.86*
MaxEnt with POSE	*94.48*	*5.52*	*97.75*	*2.25*	*97.21*
MaxEnt with POSE + Persian Specific Features	*95.92*	*4.08*	*98.18*	*1.82*	*97.80*
CRF Winsize=5	*95.15*	*4.85*	*98.36*	*1.63*	*97.83*
CRF Winsize=5 +Persian Specific Features	*96.42*	*3.58*	*98.367*	*1.63*	***98.04***

Table 4 - Comparison of results (%)

9 Conclusions

In this paper, we proposed a POSE tagging approach to recognize Ezafe in Persian sentences. Besides to this probabilistic approach, some features were extracted to increase the recognition accuracy. Experimental results show that CRF tagger acts pretty well in Persian Ezafe recognition. The obtained results show outstanding performance comparing to earlier approaches and the accuracy is quite reliable because of training based on a 10 million-words corpus.Future research can be done based on other taggers such as log-linear and TnT taggers. Moreover, Ezafe recognition can be viewed as a spell checking problem.So, a spell checker can also be used as another approach.

References

Amiri, Hadi, Hojjat, Hossein, and Oroumchian, Farhad.,2007.*Investigation on a Feasible Corpus for Persian POS Tagging*. 12th international CSI computer conference, Iran.

Bijankhan, Mahmoud.,*The Role of the Corpus in Writing a Grammar: An Introduction to a Software*, Iranian Journal of Linguistics, vol. 19, no. 2, fall and winter 2004.

Bijankhan, Mahmoud., 2005. *A feasibility study on Ezafe Domain Analysis based on pattern matching method*. Published by Research Institute on Culture, Art, and Communication, Tehran, Iran.

Estaji, Azam., Jahangiri, Nader., 2006 *The origin of kasre ezafe in persian language.* Journal of Persian language and literature, Vol. 47, pp 69-82, Isfahan University, Iran.

Farghaly, Ali., 2004. *Computer Processing of Arabic Script-based Languages: Current State and Future Directions.* Workshop on Computational Approaches to Arabic Script-based Languages, COLING 2004, University of Geneva, Geneva, Switzerland, August 28, 2004.

Ghomeshi, Jila. 1996. *Projection and Inflection: A Study of Persian Phrase Structure.* PhD. Thesis, Graduate Department of Linguistics, University of Toronto.

Isapour, Shahriyar., Homayounpour, Mohammad Mehdi, and Bijankhan, Mahmoud., 2007. *Identification of ezafe location in Persian language with Probabilistic Context Free Grammar*, 13th Computer association Conference, Kish Island, Iran.

Kahnemuyipour, Arsalan., 2003. *Syntactic categories and Persian stress.* Natural Language & Linguistic Theory 21.2: 333-379.

Lafferty, John.,McCallum, Andrew., and Pereira. Fernando, C.N., 2001 *Conditional random fields: Probabilistic models for segmenting and labeling sequence data,* In Proc. of ICML, pp.282-289,

Mavvaji, Vahid., and Eslami, Moharram., 2012. *Converting persian text to phoneme stream based on a syntactic analyser.* The first internationasl conference on persian text and speech, September 5,6, 2012, Semnan, Iran.

Namnabat, Majid., and Homayounpour, Mohamad Mehdi., 2006. *Text to phoneme conversion in Persian language using multi-layer perceptron neural network*, Iranian Journal of electrical and computer engineering, Vol. 5, No. 3, Autumn 2007.

Oskouipour, Navid., 2011. *Converting Text to phoneme stream with the ability to recognizing ezafe marker and homographs applied to Persian speech synthesis.* Msc. Thesis, Sharif University of Technology, Iran.

Pantcheva, Marina Blagoeva., 2006. *Persian Preposition Classes.* Nordlyd; Volume 33 (1). ISSN 0332-7531.s 1 - 25.

Parsafar Parviz. 2010, *Syntax, Morphology, and Semantics of Ezafe.*Iranian Studies[serial online]. December 2010;43(5):637-666. Available in Academic Search Complete, Ipswich, MA.

Razi, Behnam, and Eshqi, Mohammad, 2012. *Design of a POS tagger for Persian speech based on Neural Networks*, 20th conference on Electrical Engineering,15-17 may 2012,Tehran, Iran.

Samvelian, Pollet. 2007. *The Ezafe as a head-marking inflectional affix: Evidence from Persian and Kurmanji Kurdish.* Aspects of Iranian Linguistics: Papers in Honor of Mohammad Reza Bateni, 339-361.

Sha, Fei., and Pereira, Fernando, 2003. *Shallow parsing with conditional random fields,* In Proc. of HLT/NAACL 2003.

Shakeri, Zakieh, et al. 2012. *Use of linguistic features for improving English-Persian SMT*. Konvens 2012, The 11th Conference on Natural Language Processing, Vienna, Sept 19-21, 2012

Toutanova, Kristina Klein, and Manning. ChristopherD., 2000. *Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger*. Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000), pp. 63-70.

Toutanova, Kristina Klein, Manning, Christopher D., and Singer, Yoram. 2003. *Feature-Rich Part-of-Speech Tagging with a Cyclic Dependency Network.* In Proceedings of HLT-NAACL 2003, pp. 252-259.

Yarowsky, David. 1996. *Homograph disambiguation in text-to-speech synthesis; Progress in Speech Synthesis.* eds. van Santen, J., Sproat, R., Olive, J. and Hirschberg, J : 157-172.

Zahedi, Morteza., 1998. *Design and implementation of an intelligent program for recognizing short vowels in Persian text*. Msc. Thesis, University of Tehran.

Converting Russian Dependency Treebank to Stanford Typed Dependencies Representation

Janna Lipenkova
Institut für Deutsche und
Niederländische Philologie
Freie Universität Berlin
janna.lipenkova@fu-berlin.de

Milan Souček
Lionbridge Technologies, Inc.
Tampere, Finland
milan.soucek@lionbridge.com

Abstract

In this paper, we describe the process of rule-based conversion of Russian dependency treebank into the Stanford dependency (SD) schema. The motivation behind this project is the expansion of the number of languages that have treebank resources available in one consistent annotation schema. Conversion includes creation of Russian-specific SD guidelines, defining conversion rules from the original treebank schema into the SD model and evaluation of the conversion results. The converted treebank becomes part of a multilingual resource for NLP purposes.

1 Introduction

Dependency parsing has provided new methods and resources for natural language technology tasks in recent years. Dependency treebanks are now available for many languages and parsed data is used for improving machine translation, search engines and other NLP applications. While data resources are relatively common for monolingual tasks, there is a growing need for consistently annotated multilingual data. First larger activities in generating comparable standardized sets of multilingual parsed data were presented in CONLL shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007; Surdeanu et al., 2008; Hajič et al., 2009). More recently, cross-language consistency has been achieved by using one universal schema for all covered languages (McDonald et al., 2013). This universal treebank schema uses consistent sets of part-of-speech (POS) (Petrov et al., 2012) and dependency labels (deprel) following the Stanford typed dependencies representation (SD) (de Marneffe and Manning, 2008). The consistent treebank schema has many advantages, mainly the more straightforward possibility to build applications for multiple languages (McDonald et al., 2011), though it also presents challenges such as handling language-specific features among languages of different types without introducing conflicts. A certain level of generalization of language features that might lead to simplification is needed, as already highlighted by McDonald et al. (2013). For such universal multilingual treebank model, more resources can be built manually or they can be obtained by converting existing treebanks that follow different parsing schemas into one consistent treebank model. For the SD schema, treebanks for several languages already have been built using the manual annotation procedures (McDonald et al, 2013; Souček et al., 2013). There are also other existing treebanks covering languages from different families where the SD schema was applied (e.g. Chang et al., 2009 for Chinese; Haverinen et al., 2010 for Finnish; Seraji et al., 2012 for Persian; Tsarfaty, 2013 for Hebrew). Treebank conversion was applied e.g. in Italian (Bosco et al, 2013). The conversion model is a more suitable option for languages for which treebanks are already available, since manual annotation can be limited and the annotation/conversion process can be automated. In this paper, we describe the conversion of an existing Russian treebank

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 143–147,
Gothenburg, Sweden, April 26-30 2014.

(SynTagRus, Boguslavsky et al., 2000) into the SD schema. We present the conversion process (2), introduce the source and the target model, including adaptations of the SD schema for Russian (3), describe the conversion script (4) and finally compare the conversion results (in terms of process efficiency and output accuracy) to other tasks for which a similar process was applied (5).

2 Conversion process

The conversion procedure used in our project is similar to the process described in Bosco et al. (2013). The very first step was the development of the Russian-specific set of POS and the list of Stanford dependency relations, compliant with the standards presented by Petrov et al. (2012) and de Marneffe and Manning (2008). Next, we investigated POS and deprel sets for the original treebank and defined conversion rules for adapting data to our specific schema. A rule-based conversion script was developed on a small portion of the original data and the remaining data was automatically converted. The quality of the conversion output was monitored by manual review of samples of converted data and also using parser evaluation methods. During the manual review, errors in samples of converted data were manually corrected in order to produce a gold standard data set; at the same time further conversion rules were reported for improving the conversion script. This cycle was repeated several times until an acceptable output quality was reached.

3 Source and target models

3.1 Source model

The source data for Russian are taken from the SynTagRus treebank (Boguslavsky et al., 2000). Just as in the basic SD model, SynTagRus provides a dependency tree for each sentence where each word is connected to a head and assigned one of 78 deprels, theoretically motivated by Melcuk's Meaning-Text theory (Melcuk, 1981). Additionally, the treebank specifies POS information as well as applicable morphological information (gender, number, case, degree of comparison, aspect, tense, person, voice).

3.2 Target model

The basic version of SD (de Marneffe and Manning, 2008) counts approximately 53 dependency labels. They are used in conjunction with a "universal" set of part-of-speech labels (Petrov et al., 2012). Although our aim is to build a resource that follows a consistent schema with other existing SD languages, we decided to make some minor modifications to the SD model to account for language-specific phenomena and thus minimize the loss of structural information. Both the set of SD dependencies and of POS labels were slightly adjusted to adapt the model to Russian. All these specifics can be further converted to the fully consistent SD model. The following modifications were made to the dependencies annotation schema:

• *scomp* is introduced for the complements of (ellipted) copulas.

• *ocomp* is introduced for verb complements that are semantically predicated by the object of the verb (e.g., *I find [this idea]$_i$ interesting$_i$*.).

• *gmod* is introduced for genitive modifiers of nominals; in turn, the *poss* relation for prenominal possessive modifiers is eliminated.

• *interj* is introduced for discourse particles attaching to nominals or verbs.

Despite the modifications, the adopted model still leads to losses in more fine-grained information. An example where this becomes especially visible are objects of verbs: the SD model uses the two labels *dobj* and *iobj* for direct and indirect objects. In Russian, there is a larger range of object types; they are distinguished not only morphologically, but also syntactically (e.g. genitive of negation, whereby the negation of the verb leads to the `switch' of the direct object from accusative to genitive). In order to capture these distinctions, the original treebank uses five relations (*1-compl*, *2-compl* etc.). However, the reduction to the two types *dobj* and *iobj* assumed

for our SD model `deletes' these more fine-grained distinctions.

4 Conversion Script

4.1 General approach

The conversion script works with conversion patterns, which specify the possible targets of a source label and the conditions for this target to be applied. Conditions can be specified in terms of POS, morphological, lexical and structural information. Most conversion patterns have a regular formal basis and can be grouped into data structures that are processed by standardized functions. However, there are many patterns, especially less frequent ones, that have an irregular, more complex structure and thus have to be described individually. In order to increase the flexibility in formulating conversion patterns by specifying lexical information, the script is enriched with a set of lexical lists. These lexical lists mostly contain closed classes of functional words or idiosyncratic items, such as pronouns, subordinating conjunctions or idioms.

4.2 Conversion

Conversion acts on three types of information – POS tags, dependency relations and tree structures.

4.2.1 POS tags

The original data are annotated with five POS labels (NOUN, VERB, ADJ, ADV, CONJ); the target set contains 15 POS labels. One-to-many correspondences, for example the ambiguity of original NOUN between target NOUN and PRON, mostly occur in cases where the original POS tag subsumes some smaller class of functional words. As described above, word lists were used to identify these closed classes and to choose between the possible target POS tags.

4.2.2 Dependency relations

In the original treebank, 78 dependency relations are used; the target model contains 51 relations. For some original dependency labels, a one-to-one correspondence can be established. For example, the original label *advrb-subj*, used for nominals with an adverbial function, is always converted to *npadvmod*. However, most original dependency labels have multiple SD counterparts; conditional branching is used to determine the target relation for a given case. All types of information available in the treebank – POS, morphological, lexical and structural information – can be used to formulate conditions; in most cases, the specification for a given source relation involves a mix of the different information types.

Examples for the different types of conversion conditions are as follows:

- POS tag condition: *attrib*: convert to *nn* if NOUN, *amod* when ADJ, *prep* when ADP
- Morphological condition: *aux*: convert to *npadvmod* if in ablative case, *iobj* if in dative case.
- Structural condition: *explet*: convert to *mark* if dependent of *purpcl* or *rcmod*; *ccomp* if dependent of *complm*.
- Lexical condition: *aux*: convert to *neg* if expressed by *не/ни*, else *interj*.

4.2.3 Structural modifications

Structural modifications were introduced in several cases; most of them are caused by the reliance of SD on semantic heads and, thus, on content words as heads. During conversion, head-dependent structures are “switched” in cases where the original head does not correspond to the “semantic” head. Specifically, this occurs in the following cases:

- Structures with auxiliary verbs (future tense, passive voice): switch between auxiliary and lexical verb, so that the auxiliary depends on the lexical verb.
- Clauses introduced by subordinating conjunctions: switch between introducing conjunction and verb in the subordinate clause, so that the verb in the subordinate clause depends

no more on the conjunction, but on the verb in the matrix clause.

• Coordination structures: in the original data, the conjuncts and coordination particles form a chain, whereby each node depends on the previous one. In the target representation, all conjuncts and coordination particles attach to the first conjunct.

4.2.4 Problems and inaccuracies - syntactic under-specification

Under the SD model, different dependency relations may apply to structurally identical, but semantically different relations. For example, postverbal nominals in instrumental case can be either *iobj* (indirect object, corresponding to the instrument argument) or *npadvmod* (nominal adverbial); the relation applicable in a given case depends on the lexical semantics of the verb and the nominal:

(1) a. *npadvmod*(gaze, wolf):
смотреть *волком*
gaze wolf.INS
'to gaze angrily'
b. *iobj*(cut, knife):
резать *ножом*
cut knife.INS
'to cut with a knife'

The semantic difference is not visible at a surface level: there is no structural criterion which might condition the choice of the target relation. Since both structures are lexically productive, basing the choice on word lists is also not a satisfactory solution. Rather, the disambiguation of these and similar cases would require a more fine-grained semantic classification specifying valence frames and selectional restrictions of verbs as well as semantic features of nouns; in example (1), such a classification would allow to identify verbs that semantically select instruments (corresponding to *iobj*) as well as nouns that can potentially act as instruments. Besides, machine learning techniques can also be used for disambiguation based on the frequency of the lexical constellations for a particular dependency relation. Another problem are non-frequent dependency relations and contexts of occurrence which do not provide enough evidence for postulating a reliable, universally applicable conversion pattern. In the original treebank, 24 out of 78 dependency relations have a frequency of occurrence of less than 100. Besides, after the application of the conversion patterns, numerous dependency relations remain non-converted, because their contexts of occurrence are non-frequent and thus also cannot be reliably captured by conversion patterns. Our model uses the generic label *xdep* to identify tokens for which conversion was not successful. This label mostly appears for tokens whose original deprels do not allow for a rule-based characterization because they are partially defined in semantic terms, such as *nonself-agent*, *distrib*, *elaborat* and *mod-descr*.

5 Results

The presented script converts the original Russian treebank fully into the SD schema. The converted treebank data is owned by Google and its availability can be checked with the data owners. Conversion output precision was measured with MaltEval (Nilsson and Nivre, 2008) using manually annotated 500 sentences as gold standard and the same set processed with the conversion script as a test data. We achieved 76.21% LAS and 83.84% UAS. Achieved LAS is slightly lower than for similar work reported for Italian (Bosco et al., 2013), where LAS for different sub-models is between 79.94% and 84.14% in the parser output. Since the aim of this project is to create comparable cross-language data with acceptable precision within reasonable time frame, the precision that we achieved seems to be in acceptable range for the described conversion task.

6 Conclusions and future work

Our further target is to build similar conversion tasks for other languages, where existing treebanks are available. We also plan to take advantage of machine learning mechanisms that can make the conversion work more efficient.

Acknowledgemnts

This work was performed by Lionbridge NLS team for Google. We would like to thank Google for giving us the opportunity to work on this project and to publish our results.

References

Igor Boguslavsky; Svetlana Grigorieva; Nikolai Grigoriev; Leonid Kreidlin; Nadezhda Frid. 2000. *Dependency Treebank for Russian: Concept, Tools, Types of Information.* In: COLING 2000 Volume 2.

Cristina Bosco, Simonetta Montemagni, Maria Simi. 2013. *Converting Italian Treebanks: Towards an Italian Stanford Dependency Treebank*. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL).

Sabine Buchholz and Erwin Marsi. 2006. *CoNLL X shared task on multilingual dependency parsing*. In Proceedings of CoNLL.

Pi-Chuan Chang, Huihsin Tseng, Dan Jurafsky, and Christopher D. Manning. 2009. *Discriminative reordering with Chinese grammatical relations features.* In Proceedings of the Third Workshop on Syntax and Structure in Statistical Translation (SSST-3) at NAACL HLT 2009.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria Antónia Martí, Lluís Márquez, Adam Meyers, Joakim Nivre, Sebastian Padó, Jan Štěpánek, Pavel Straňák, Mihai Surdeanu, Nianwen Xue, Yi Zhang. 2009. *The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages*. In Proceedings of CoNLL.

Katri Haverinen, Timo Viljanen, Veronika Laippala, Samuel Kohonen, Filip Ginter, and Tapio Salakoski. 2010. *Treebanking Finnish*. In Proceedings of TLT9, pp. 79–90

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. *Stanford typed dependencies manual.*

Ryan McDonald, Slav Petrov, and Keith Hall. 2011. *Multi-source transfer of delexicalized dependency parsers*. In Proceedings of EMNLP.

Ryan McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. *Universal Dependency Annotation for Multilingual Parsing.* In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL).

Igor Melcuk. 1981. Meaning-Text Models: A recent Trend in Soviet Linguistics. Annual Review of Anthropology 10, 27–62.

Jens Nilsson, and Joakim Nivre. 2008. MaltEval: An Evaluation and Visualization Tool for Dependency Parsing. In Proceedings of LREC.

Joakim Nivre, Johan Hall, Sandra Kübler, Ryan McDonald, Jens Nilsson, Sebastian Riedel, and Deniz Yuret. 2007. *The CoNLL 2007 shared task on dependency parsing*. In Proceedings of EMNLPCoNLL.

Slav Petrov, Dipanjan Das, and Ryan McDonald. 2012. A universal part-of-speech tagset. In Proceedings of LREC.

Mojgan Seraji, Be´ata Megyesi, and Nivre Joakim. 2012. *Bootstrapping a Persian dependency treebank*. Linguistic Issues in Language Technology, 7.

Milan Souček, Timo Järvinen, Adam LaMontagne. 2013. *Managing a Multilingual Treebank Project.* In Proceedings of the Second International Conference on Dependency Linguistics.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluís Màrquez, and Joakim Nivre. 2008. *The CoNLL-2008 shared task on joint parsing of syntactic and semantic dependencies*. In Proceedings of CoNLL.

Reut Tsarfaty. 2013. *A unified morpho-syntactic scheme of Stanford dependencies*. Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL).

Integrating an Unsupervised Transliteration Model into Statistical Machine Translation

Nadir Durrani
University of Edinburgh
dnadir@inf.ed.ac.uk

Hassan Sajjad
Qatar Computing Research Institute
hsajjad@@qf.org.qa

Hieu Hoang **Philipp Koehn**
University of Edinburgh
hieu.hoang,pkoehn@inf.ed.ac.uk

Abstract

We investigate three methods for integrating an unsupervised transliteration model into an end-to-end SMT system. We induce a transliteration model from parallel data and use it to translate OOV words. Our approach is fully unsupervised and language independent. In the methods to integrate transliterations, we observed improvements from 0.23-0.75 (Δ 0.41) BLEU points across 7 language pairs. We also show that our mined transliteration corpora provide better rule coverage and translation quality compared to the gold standard transliteration corpora.

1 Introduction

All machine translation (MT) systems suffer from the existence of out-of-vocabulary (OOV) words, irrespective of the amount of data available for training. OOV words are mostly named entities, technical terms or foreign words that can be translated to the target language using transliteration.

Much work (Al-Onaizan and Knight, 2002; Zhao et al., 2007; Kashani et al., 2007; Habash, 2009) has been done on transliterating named entities and OOVs, and transliteration has been shown to improve MT quality. Transliteration has also shown to be useful for translating closely related language pairs (Durrani et al., 2010; Nakov and Tiedemann, 2012), and for disambiguation (Hermjakob et al., 2008; Azab et al., 2013). However, despite its utility, a transliteration module does not exist in the commonly used MT toolkits, such as Moses (Koehn et al., 2007). One of the main reasons is that the training data, a corpus of transliteration pairs, required to build a transliteration system, is not readily available for many language pairs. Even if such a training data is available, mechanisms to integrate transliterated words into MT pipelines are unavailable in these toolkits. Generally, a supervised transliteration system is trained separately outside of an MT pipeline, and a naïve approach, to replace OOV words with their 1-best transliterations in the post/pre-processing step of decoding is commonly used.

In this work i) we use an unsupervised model based on Expectation Maximization (EM) to induce transliteration corpus from word aligned parallel data, which is then used to train a transliteration model, ii) we investigate three different methods for integrating transliteration during decoding, that we implemented within the Moses toolkit. To the best of our knowledge, our work is the foremost attempt to integrate unsupervised transliteration model into SMT.

This paper is organized as follows. Section 2 describes the unsupervised transliteration mining system, which automatically mines transliteration pairs from the same word-aligned parallel corpus as used for training the MT system. Section 3 describes the transliteration model that is trained using the automatically extracted pairs. Section 4 presents three methods for incorporating transliteration into the MT pipeline, namely: i) replacing OOVs with the 1-best transliteration in a post-decoding step, ii) selecting the best transliteration from the list of n-best transliterations using transliteration and language model features in a post-decoding step, iii) providing a transliteration phrase-table to the decoder on the fly where it can consider all features to select the best transliteration of OOV words. Section 5 presents results. Our integrations achieved an average improvement of 0.41 BLEU points over a competitive baseline across 7 language pairs (Arabic, Bengali, Farsi, Hindi, Russian, Telugu and Urdu-into-English). An additional experiment showed that our system provides better rule coverage as opposed to another built from gold standard transliteration corpus and produces better translations.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 148–153,
Gothenburg, Sweden, April 26-30 2014.

2 Transliteration Mining

The main bottleneck in building a transliteration system is the lack of availability of transliteration training pairs. It is, however, fair to assume that any parallel data would contain a reasonable number of transliterated word pairs. Transliteration mining can be used to extract such word pairs from the parallel corpus. Most previous techniques on transliteration mining generally use supervised and semi-supervised methods (Sherif and Kondrak, 2007; Jiampojamarn et al., 2010; Darwish, 2010; Kahki et al., 2012). This constrains the mining solution to language pairs for which training data (seed data) is available. A few researchers proposed unsupervised approaches to mine transliterations (Lee and Choi, 1998; Sajjad et al., 2011; Lin et al., 2011). We adapted the work of Sajjad et al. (2012) as summarized below.

Model: The transliteration mining model is a mixture of two sub-models, namely: a transliteration and a non-transliteration sub-model. The idea is that the transliteration model would assign higher probabilities to transliteration pairs compared to the probabilities assigned by a non-transliteration model to the same pairs. Consider a word pair (e, f), the **transliteration model** probability for the word pair is defined as follows:

$$p_{tr}(e,f) = \sum_{a \in Align(e,f)} \prod_{j=1}^{|a|} p(q_j)$$

where $Align(e, f)$ is the set of all possible sequences of character alignments, a is one alignment sequence and q_j is a character alignment.

The **non-transliteration model** deals with the word pairs that have no character relationship between them. It is modeled by multiplying source and target character unigram models:

$$p_{ntr}(e,f) = \prod_{i=1}^{|e|} p_E(e_i) \prod_{i=1}^{|f|} p_F(f_i)$$

The **transliteration mining model** is defined as an interpolation of the transliteration sub-model and the non-transliteration sub-model:

$$p(e,f) = (1-\lambda) p_{tr}(e,f) + \lambda p_{ntr}(e,f)$$

λ is the prior probability of non-transliteration.

The non-transliteration model does not change during training. We compute it in a pre-processing step. The transliteration model learns character alignment using expectation maximization (EM). See Sajjad et al. (2012) for more details.

3 Transliteration Model

Now that we have transliteration word pairs, we can learn a transliteration model. We segment the training corpus into characters and learn a phrase-based system over character pairs. The transliteration model assumes that source and target characters are generated monotonically.[1] Therefore we do not use any reordering models. We use 4 basic phrase-translation features (direct, inverse phrase-translation, and lexical weighting features), language model feature (built from the target-side of mined transliteration corpus), and word and phrase penalties. The feature weights are tuned[2] on a dev-set of 1000 transliteration pairs.

4 Integration to Machine Translation

We experimented with three methods for integrating transliterations, described below:

Method 1: involves replacing OOVs in the output with the 1-best transliteration. The success of Method 1 is solely contingent on the accuracy of the transliteration model. Also, it ignores context which may lead to incorrect transliteration. For example, the Arabic word بيل transliterates to "Bill" when followed by "Clinton" and "Bell" if preceded by "Alexander Graham".

Method 2: provides n-best transliterations to a monotonic decoder that uses a monolingual language model and a transliteration phrase-translation table to rescore transliterations. We carry forward the 4 translation model features used in the transliteration system to build a transliteration phrase-table. We additionally use an LM-OOV feature which counts the number of words in a hypothesis that are unknown to the language model. Smoothing methods such as Kneser-Ney assign significant probability mass to unseen events, which may cause the decoder to make incorrect transliteration selection. The LM-OOV feature acts as a prior to penalize such hypotheses.

Method 3: Method 2 can not benefit from all in-decoding features and phenomenon like reordering. It transliterates Urdu compound بحیرہ عرب (Arabian Sea) to "Sea Arabian", if عرب is an unknown word. In method 3, we feed the transliteration phrase-table directly into the first-pass decoding which allows reordering of UNK words. We

[1] Mining algorithm also makes this assumption.

[2] Tuning data is subtracted from the training corpus while tuning to avoid over-fitting. After the weights are tuned, we add it back, retrain GIZA, and estimate new models.

use the *decoding-graph-backoff* option in Moses, that allows multiple translation phrase tables and back-off models. As in method 2, we also use the LM-OOV feature in method 3.[3]

5 Evaluation

Data: We experimented with 7 language pairs, namely: Arabic, Bengali, Farsi, Hindi, Russian, Telugu and Urdu-into-English. For Arabic[4] and Farsi, we used the TED talks data (Cettolo et al., 2012) made available for IWSLT-13, and we used the dev2010 set for tuning and the test2011 and test2012 sets for evaluation. For Indian languages we used the Indic multi-parallel corpus (Post et al., 2012), and we used the dev and test sets provided with the parallel corpus. For Russian, we used WMT-13 data (Bojar et al., 2013), and we used half of the news-test2012 for tuning and other half for testing. We also evaluated on the news-test2013 set. For all, we trained the language model using the monolingual WMT-13 data. See Table 1 for data statistics.

Lang	$\mathbf{Train_{tm}}$	$\mathbf{Train_{tr}}$	Dev	$\mathbf{Test_1}$	$\mathbf{Test_2}$
AR	152K	6795	887	1434	1704
BN	24K	1916	775	1000	
FA	79K	4039	852	1185	1116
HI	39K	4719	1000	1000	
RU	2M	302K	1501	1502	3000
TE	45K	4924	1000	1000	
UR	87K	9131	980	883	

Table 1: No. of sentences in Training Data and Mined Transliteration Corpus (Types) ($\mathbf{Train_{tr}}$)

Baseline Settings: We trained a Moses system replicating the settings used in competition-grade systems (Durrani et al., 2013b; Birch et al., 2013): a maximum sentence length of 80, GDFA symmetrization of GIZA++ alignments (Och and Ney, 2003), an interpolated Kneser-Ney smoothed 5-gram language model with KenLM (Heafield, 2011) used at runtime, a 5-gram OSM (Durrani et al., 2013a), msd-bidirectional-fe lexicalized reordering, sparse lexical and domain features (Hasler et al., 2012), a distortion limit of 6, 100-best translation options, MBR decoding (Kumar and Byrne, 2004), Cube Pruning (Huang and Chiang, 2007), and the no-reordering-over-punctuation heuristic. We tuned with the k-best batch MIRA (Cherry and Foster, 2012).[5]

Transliteration Miner: The miner extracts transliterations from a word-aligned parallel corpus. We only used word pairs with 1-to-1 alignments.[6] Before feeding the list into the miner, we cleaned it by removing digits, symbols, word pairs where source or target is composed from less than 3 characters, and words containing foreign characters that do not belong to this scripts. We ran the miner with 10 iterations of EM. The number of transliteration pairs (types) extracted for each language pair is shown in Table 1 ($\mathbf{Train_{tr}}$).

Transliteration System: Before evaluating our integrations into the SMT system, we performed an intrinsic evaluation of the transliteration system that we built from the mined pairs. We formed test data for Arabic–English (1799 pairs), Hindi–English (2394 pairs) and Russian–English (1859 pairs) by concatenating the seed data and gold standard transliteration pairs both provided for the Shared Task on Transliteration mining (Kumaran et al., 2010). Table 2 shows precision and recall of the mined transliteration system (MTS).

	AR	HI	RU
Precision (1-best Accuracy)	20.0%	25.3%	46.1%
Recall (100-best Accuracy)	80.2%	79.3%	87.5%

Table 2: Precision and Recall of MTS

The precision (1-best accuracy) of the transliteration model is quite low. This is because the transliteration corpus is noisy and contains imperfect transliteration pairs. For example, the miner extracted the pair (استراليا, Australasia), while the correct transliteration is "Australia". We can improve the precision by tightening the mining threshold probability. However, our end goal is to improve end-to-end MT and not the transliteration system. We observed that recall is more important than precision for overall MT quality. We provide an empirical justification for this when discussing the final experiments.

[3]Method 3 is desirable in cases where the decoder can translate or transliterate a word. For example Hindi word सीमा can be translated to "Border" and also transliterated to name "Seema". Identifying such candidates that can be translated or transliterated is a challenge. Machine learning techniques (Goldwasser and Roth, 2008; Kirschenbaum and Wintner, 2009) and named entity recognizers (Klementiev and Roth, 2006; Hermjakob et al., 2008) have been used for this purpose. Though, we only focus on OOV words, method 3 can be used if such a classifier/NE tagger is available.

[4]Arabic and Urdu are segmented using MADA (Habash and Sadat, 2006) and UWS (Durrani and Hussain, 2010).

[5]Retuning the transliteration features was not helpful, default weights are used.

[6]M-N/1-N alignments are less likely to be transliterations.

MT Experiments: Table 3 gives a comprehensive evaluation of the three methods of integration discussed in Section 4 along with the number[7] of OOV words (types) in different tests. We report BLEU gains (Papineni et al., 2002) obtained by each method. Method 1 (M_1), that replaces OOV words with 1-best transliteration gave an average improvement of +0.13. This result can be attributed to the low precision of the transliteration system (Table 2). Method 2 (M_2), that transliterates OOVs in second pass monotonic decoding, gave an average improvement of +0.39. Slightly higher gains were obtained using Method 3 (M_3), that integrates transliteration phrase-table inside decoder on the fly. However, the efficacy of M_3 in comparison to M_2 is not as apparent, as M_2 produced better results than M_3 in half of the cases.

Lang	Test	B_0	M_1	M_2	M_3	OOV
AR	$iwslt_{11}$	26.75	+0.12	+0.36	+0.25	587
	$iwslt_{12}$	29.03	+0.10	+0.30	+0.27	682
BN	jhu_{12}	16.29	+0.12	+0.42	+0.46	1239
FA	$iwslt_{11}$	20.85	+0.10	+0.40	+0.31	559
	$iwslt_{12}$	16.26	+0.04	+0.20	+0.26	400
HI	jhu_{12}	15.64	+0.21	+0.35	+0.47	1629
RU	wmt_{12}	33.95	+0.24	+0.55	+0.49	434
	wmt_{13}	25.98	+0.25	+0.40	+0.23	799
TE	jhu_{12}	11.04	-0.09	+0.40	+0.75	2343
UR	jhu_{12}	23.25	+0.24	+0.54	+0.60	827
Avg		21.9	+0.13	+0.39	+0.41	950

Table 3: End-to-End MT Evaluation – B_0 = Baseline, M_1 = $Method_1$, M_2 = $Method_2$, M_3 = $Method_3$, BLEU gains shown for each method

In an effort to test whether improving transliteration precision would improve end-to-end SMT results, we carried out another experiment. Instead of building a transliteration system from mined corpus, we built it using the gold standard corpus (for Arabic, Hindi and Russian), that we also used previously to do an intrinsic evaluation. We then replaced our mined transliteration systems with the gold standard transliteration systems, in the best performing SMT systems for these languages. Table 4 shows a comparison of performances. Although the differences are small, systems using mined transliteration system (MTS) outperformed its counterpart that uses gold standard transliteration system (GTS), except in Hindi–English where both systems were equal.

[7]Note that not all OOVs can be transliterated. This number is therefore an upper bound what can be transliterated.

	AR		HI	RU	
	$iwslt_{11}$	$iwslt_{12}$	jhu_{12}	wmt_{12}	$iwslt_{13}$
MTS	27.11	29.33	16.11	34.50	26.38
GST	26.99	29.20	16.11	34.33	26.22

Table 4: Comparing Gold Standard Transliteration (GST) and Mined Transliteration Systems

In the error analysis we found that the GST system suffered from sparsity and did not provide enough coverage of rules to produce right transliterations. For example, Arabic drops the determiner ال (al), but such additions were not observed in gold transliteration pairs. Arabic word الغيغابكسل (Gigapixel) is therefore transliterated to "**al**gegabksl". Similarly the GST system learned no transliteration pairs to account for the rule "b → p" and therefore erroneously transliterated سبرلوك (Spurlock) to "Sbrlok". Similar observations were true for the case of Russian–English. The rules "a → u" and "y → ϵ" were not observed in the gold set, and hence харрикейнз (hurricane) was transliterated to "herricane" and талботу (Talbot) to "Talboty". This shows that better recall obtained from the mined pairs led to overall improvement.

6 Conclusion

We incorporated unsupervised transliteration mining model into standard MT pipeline to automatically transliterate OOV words without needing additional resources. We evaluated three methods for integrating transliterations on 7 language pairs and showed improvements ranging from 0.23-0.75 (Δ 0.41) BLEU points. We also showed that our mined transliteration corpus provide better recall and overall translation quality compared to the gold standard transliteration corpus. The unsupervised transliteration miner and its integration to SMT has been made available to the research community via the Moses toolkit.

Acknowledgments

We wish to thank the anonymous reviewers and Kareem Darwish for their valuable feedback on an earlier draft of this paper. The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 287658. This publication only reflects the authors' views.

References

Yaser Al-Onaizan and Kevin Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*.

Mahmoud Azab, Houda Bouamor, Behrang Mohit, and Kemal Oflazer. 2013. Dudley North visits North London: Learning When to Transliterate to Arabic. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 439–444, Atlanta, Georgia, June. Association for Computational Linguistics.

Alexandra Birch, Nadir Durrani, and Philipp Koehn. 2013. Edinburgh SLT and MT System Description for the IWSLT 2013 Evaluation. In *Proceedings of the 10th International Workshop on Spoken Language Translation*, pages 40–48, Heidelberg, Germany, December.

Ondrej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Eighth Workshop on Statistical Machine Translation*, WMT-2013, pages 1–44, Sofia, Bulgaria.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT^3: Web Inventory of Transcribed and Translated Talks. In *Proceedings of the 16th Conference of the European Association for Machine Translation (EAMT)*, pages 261–268, Trento, Italy, May.

Colin Cherry and George Foster. 2012. Batch Tuning Strategies for Statistical Machine Translation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 427–436, Montréal, Canada, June. Association for Computational Linguistics.

Kareem Darwish. 2010. Transliteration Mining with Phonetic Conflation and Iterative Training. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.

Nadir Durrani and Sarmad Hussain. 2010. Urdu Word Segmentation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 528–536, Los Angeles, California, June. Association for Computational Linguistics.

Nadir Durrani, Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2010. Hindi-to-Urdu Machine Translation through Transliteration. In *Proceedings of the 48th Annual Conference of the Association for Computational Linguistics*, Uppsala, Sweden.

Nadir Durrani, Alexander Fraser, Helmut Schmid, Hieu Hoang, and Philipp Koehn. 2013a. Can Markov Models Over Minimal Translation Units Help Phrase-Based SMT? In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, Sofia, Bulgaria, August. Association for Computational Linguistics.

Nadir Durrani, Barry Haddow, Kenneth Heafield, and Philipp Koehn. 2013b. Edinburgh's Machine Translation Systems for European Language Pairs. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, Sofia, Bulgaria, August. Association for Computational Linguistics.

Dan Goldwasser and Dan Roth. 2008. Active Sample Selection for Named Entity Transliteration. In *Proceedings of ACL-08: HLT, Short Papers*, pages 53–56, Columbus, Ohio, June. Association for Computational Linguistics.

Nizar Habash and Fatiha Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52, New York City, USA, June. Association for Computational Linguistics.

Nizar Habash. 2009. REMOOV: A Tool for Online Handling of Out-of-Vocabulary Words in Machine Translation. In *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, April. The MEDAR Consortium.

Eva Hasler, Barry Haddow, and Philipp Koehn. 2012. Sparse Lexicalised Features and Topic Adaptation for SMT. In *Proceedings of the seventh International Workshop on Spoken Language Translation (IWSLT)*, pages 268–275.

Kenneth Heafield. 2011. KenLM: Faster and Smaller Language Model Queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197, Edinburgh, Scotland, United Kingdom, 7.

Ulf Hermjakob, Kevin Knight, and Hal Daumé III. 2008. Name Translation in Statistical Machine Translation - Learning When to Transliterate. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Columbus, Ohio.

Liang Huang and David Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 144–151, Prague, Czech Republic, June. Association for Computational Linguistics.

Sittichai Jiampojamarn, Kenneth Dwyer, Shane Bergsma, Aditya Bhargava, Qing Dou, Mi-Young Kim, and Grzegorz Kondrak. 2010. Transliteration

Generation and Mining with Limited Training Resources. In *Proceedings of the 2010 Named Entities Workshop*, Uppsala, Sweden.

Ali El Kahki, Kareem Darwish, Ahmed Saad El Din, and Mohamed Abd El-Wahab. 2012. Transliteration Mining Using Large Training and Test Sets. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL HLT '12.

Mehdi M. Kashani, Eric Joanis, Roland Kuhn, George Foster, and Fred Popowich. 2007. Integration of an Arabic Transliteration Module into a Statistical Machine Translation System. In *Proceedings of the Second Workshop on Statistical Machine Translation*, Prague, Czech Republic.

Amit Kirschenbaum and Shuly Wintner. 2009. Lightly Supervised Transliteration for Machine Translation. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 433–441, Athens, Greece, March. Association for Computational Linguistics.

Alexandre Klementiev and Dan Roth. 2006. Named entity transliteration and discovery from multilingual comparable corpora. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 82–88, New York City, USA, June. Association for Computational Linguistics.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open Source Toolkit for Statistical Machine Translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, Demonstration Program*, Prague, Czech Republic.

Shankar Kumar and William J. Byrne. 2004. Minimum Bayes-Risk Decoding for Statistical Machine Translation. In *HLT-NAACL*, pages 169–176.

A Kumaran, Mitesh M. Khapra, and Haizhou Li. 2010. Whitepaper of news 2010 shared task on transliteration mining. In *Proceedings of the 2010 Named Entities Workshop*, pages 29–38, Uppsala, Sweden, July. Association for Computational Linguistics.

Jae-Sung Lee and Key-Sun Choi. 1998. English to Korean Statistical Transliteration for Information Retrieval. *Computer Processing of Oriental Languages*, 12(1):17–37.

Wen-Pin Lin, Matthew Snover, and Heng Ji. 2011. Unsupervised Language-Independent Name Translation Mining from Wikipedia Infoboxes. In *Proceedings of the First workshop on Unsupervised Learning in NLP*, pages 43–52, Edinburgh, Scotland, July. Association for Computational Linguistics.

Preslav Nakov and Jörg Tiedemann. 2012. Combining Word-Level and Character-Level Models for Machine Translation Between Closely-Related Languages. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 301–305, Jeju Island, Korea, July. Association for Computational Linguistics.

Franz J. Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1).

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Morristown, NJ, USA.

Matt Post, Chris Callison-Burch, and Miles Osborne. 2012. Constructing Parallel Corpora for Six Indian Languages via Crowdsourcing. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 401–409, Montréal, Canada, June. Association for Computational Linguistics.

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2011. An Algorithm for Unsupervised Transliteration Mining with an Application to Word Alignment. In *Proceedings of the 49th Annual Conference of the Association for Computational Linguistics*, Portland, USA.

Hassan Sajjad, Alexander Fraser, and Helmut Schmid. 2012. A Statistical Model for Unsupervised and Semi-supervised Transliteration Mining. In *Proceedings of the 50th Annual Conference of the Association for Computational Linguistics*, Jeju, Korea.

Tarek Sherif and Grzegorz Kondrak. 2007. Bootstrapping a Stochastic Transducer for Arabic-English Transliteration Extraction. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic.

Bing Zhao, Nguyen Bach, Ian Lane, and Stephan Vogel. 2007. A Log-Linear Block Transliteration Model based on Bi-Stream HMMs. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, New York.

Improving Dependency Parsers with Supertags

Hiroki Ouchi **Kevin Duh** **Yuji Matsumoto**
Computational Linguistics Laboratory
Nara Institute of Science and Technology
{ouchi.hiroki.nt6, kevinduh, matsu}@is.naist.jp

Abstract

Transition-based dependency parsing systems can utilize rich feature representations. However, in practice, features are generally limited to combinations of lexical tokens and part-of-speech tags. In this paper, we investigate richer features based on supertags, which represent lexical templates extracted from dependency structure annotated corpus. First, we develop two types of supertags that encode information about head position and dependency relations in different levels of granularity. Then, we propose a transition-based dependency parser that incorporates the predictions from a CRF-based supertagger as new features. On standard English Penn Treebank corpus, we show that our supertag features achieve parsing improvements of 1.3% in unlabeled attachment, 2.07% root attachment, and 3.94% in complete tree accuracy.

1 Introduction

One significant advantage of transition-based dependency parsing (Yamada and Matsumoto, 2003; Nivre et al, 2007, Goldberg and Elhadad, 2010; Huang and Sagae, 2010) is that they can utilize rich feature representations. However, in practice, current state-of-the-art parsers generally utilize only features that are based on lexical tokens and part-of-speech (POS) tags. In this paper, we argue that more complex features that capture fine-grained syntactic phenomenon and long-distance dependencies represent a simple and effective way to improve transition-based dependency parsers.

We focus on defining supertags for English dependency parsing. Supertags, which are lexical templates extracted from dependency structure annotated corpus, encode linguistically rich information that imposes complex constraints in a local context (Bangalore and Joshi, 1999). While supertags have been used in frameworks based on lexicalized grammars, e.g. Lexicalized Tree-Adjoining Grammar (LTAG), Head-driven Phrase Structure Grammar (HPSG) and Combinatory Categorial Grammar (CCG), they have scarcely been utilized for dependency parsing so far.

Previous work by Foth et al (2006) demonstrate that supertags improve German dependency parsing under a Weighted Constraint Dependency Grammar (WCDG). Recent work by Ambati et al (2013) show that supertags based on CCG lexicon improves transition-based dependency parsing for Hindi. In particular, they argue that supertags can improve long distance dependencies (e.g. coordination, relative clause) in a morphologically-rich free-word-order language. Zhang et. al. (2010) define supertags that incorporate that long-distance dependency information for the purpose of HPSG parsing. All these works suggest the promising synergy between dependency parsing and supertagging. Our main contributions are: (1) an investigation of supertags that work well for English dependency parsing, and (2) a novel transition-based parser that effectively utilizes such supertag features.

In the following, we first describe our supertag design (Section 2) and parser (Section 3). Supertagging and parsing experiments on the Penn Treebank (Marcus et al., 1993) are shown in Section 4. We show that using automatically predicted supertags, our parser can achieve improvements of 1.3% in unlabeled attachment, 2.07% root attachment, and 3.94% in complete tree accuracy.

2 Supertag Design

The main challenge with designing supertags is finding the right balance between granularity and predictability. Ideally, we would like to increase the granularity of the supertags in order capture

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 154–158,
Gothenburg, Sweden, April 26-30 2014.

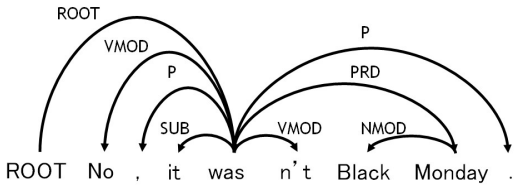


Figure 1: Example sentence

Word	Model 1	Model 2
No	VMOD/R	VMOD/R
,	P/R	P/R
it	SUB/R	SUB/R
was	ROOT+L_R	ROOT+SUB/L_PRD/R
n't	VMOD/L	VMOD/L
Black	NMOD/R	NMOD/R
Monday	PRD/L+L	PRD/L+L
.	P/L	P/L

Table 1: Model 1 & 2 supertags for Fig. 1.

more fine-grained syntactic information, but large tagsets tend to be more difficult to predict automatically. We describe two supertag designs with different levels of granularity in the following, focusing on incorporating syntactic features that we believe are important for dependency parsing.

For easy exposition, consider the example sentence in Figure 1. Our first supertag design, Model 1, represents syntactic information that shows the relative position (direction) of the head of a word, such as left (L) or right (R). If a word has root as its head, we consider it as no direction. In addition, dependency relation labels of heads are added. For instance, 'No' in the example in Figure 1 has its head in the right direction with a label 'VMOD', so its supertag can be represented as 'VMOD/R'. This kind of information essentially provides clues about the role of the word in sentence.

On top of this, we also add information about whether a word has any left or right dependents. For instance, the word 'Monday' has a left dependent 'Black', so we encode it as 'PRD/L+L', where the part before '+' specifies the head information ('PRD/L') and the part afterwards ('L') specifies the position of the dependent ('L' for left, 'R' for right). When a word has its dependents in both left and right directions, such as the word 'was' in Figure 1, we combine them using '_', as in: 'ROOT+L_R'. On our Penn Treebank data, Model 1 has 79 supertags.

unigrams of supertags	
for p in p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}, p_{i+3}	$w_p s_p$, $t_p s_p$
bigrams of supertags	
for p, q in (p_i, p_{i+1}), (p_i, p_{i+2}), (p_{i-1}, p_i), (p_{i-1}, p_{i+2}), (p_{i+1}, p_{i+2})	$s_p s_q$, $t_p s_q$, $s_p t_q$, $w_p s_q$, $s_p w_q$
head-dependent of supertags	
for p, q in (p_i, p_{i+1}), (p_i, p_{i+2}), (p_{i-1}, p_i), (p_{i-1}, p_{i+2}), (p_{i+1}, p_{i+2})	$w_p sh_p w_q sld_q$, $t_p sh_p t_q sld_q$, $w_p srd_p w_q sh_q$, $t_p srd_p t_q sh_q$

Table 2: Proposed supertag feature templates. w = word; t = POS-tag; s = supertag; sh = head part of supertag; sld = left dependent part of supertag; srd = right dependent part of supertag

In Model 2, we further add dependency relation labels of obligatory dependents of verbs. Here we define obligatory dependents of verbs as dependents which have the following dependency relation labels, 'SUB', 'OBJ', 'PRD' and 'VC'. If a label of a dependent is not any of the obligatory dependent labels, the supertag encodes only the information of direction of the dependents (same as Model 1). For instance, 'was' in the example sentence has an obligatory dependent with a label 'SUB' in the left direction and 'PRD' in the right direction, so its supertag is represented as 'ROOT+SUB/L_PRD/R'. If a verb has multiple obligatory dependents in the same direction, its supertag encodes them in sequence; if a verb takes a subject and two objects, we may have 'X/X+SUB/L_OBJ/R_OBJ/R'. The number of supertags of Model 2 is 312.

Our Model 2 is similar to Model F of Foth et al. (2006) except that they define objects of prepositions and conjunctions as obligatory as well as verbs. However, we define only dependents of verbs because verbs play the most important role for constructing syntactic trees and we would like to decrease the number of supertags.

3 Supertags as Features in a Transition-based Dependency Parser

In this work, we adopt the Easy-First parser of (Goldberg and Elhadad, 2010), a highly-accurate transition-based dependency parser. We describe how we incorporate supertag features in the Easy-First framework, though it can be done similarly

for other transition-based frameworks like left-to-right *arc-eager* and *arc-standard* models (Nivre et al., 2006; Yamada and Matsumoto, 2003).

In the Easy-First algorithm, a dependency tree is constructed by two kinds of actions: ATTACHLEFT(i) and ATTACHRIGHT(i) to a list of partial tree structures p_1,...,p_k initialized with the n words of the sentence w_1,...,w_n. ATTACHLEFT(i) attaches (p_i, p_{+1}) and removes p_{i+1} from the partial tree list. ATTACHRIGHT(i) attaches (p_{i+1}, p_i) and removes p_i from the partial tree list. Features are extracted from the attachment point as well as two neighboring structures: p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}, p_{i+3}. Table 2 summarizes the supertag features we extract from this neighborhood; these are appended to the original baseline features based on POS/word in Goldberg and Elhadad (2010).

For a partial tree structure p, features are defined based on information in its head: we use w_p to refer to the surface word form of the head word of p, t_p to refer to the head word's POS tag, and s_p to refer to the head word's supertag. Further, we not only use a supertag as is, but split each supertag into subparts. For instance, the supertag 'ROOT+SUB/L_PRD/R' is split into 'ROOT', 'SUB/L' and 'PRD/R', a supertag representing the supertag head information sh_p, supertag left dependent information sld_p, and supertag right dependent information srd_p.

For the unigram features, we use information within a single partial structure, such as conjunction of head word and its supertag ($w_p s_p$), conjunction of head word's POS tag and its supertag ($t_p s_p$). To consider more context, bigram features look at pairs of partial structures. For each (p, q) pair of structures in p_{i-2}, p_{i-1}, p_i, p_{i+1}, p_{i+2}, we look at e.g. conjunctions of supertags ($s_p s_q$).

Finally, head information of a partial structure and dependent information of another partial structure are combined as "head-dependent features" in order to check for consistency in head-dependent relations. For instance, in Table 1 the supertag for the word 'Black' has head part 'NMOD/R' wanting to attach right and the supertag for the word 'Monday' has dependent part 'L' wanting something to the left; they are likely to be attached by our parser because of the consistency in head-dependent direction. These features are used in conjunction with word and POS-tag.

Model	# tags	Dev	Test
Model1	79	87.81	88.12
Model2	312	87.22	87.13

Table 3: Supertag accuracy evaluated on development and test set. Dev = development set, PTB 22; Test = test set, PTB 23

4 Experiments

To evaluate the effectiveness of supertags as features, we perform experiments on the Penn Treebank (PTB), converted into dependency format with Penn2Malt[1]. Adopting standard approach, we split PTB sections 2-21 for training, section 22 for development and 23 for testing. We assigned POS-tags to the training data by ten-fold jackknifing following Huang and Sagae (2010). Development and test sets are automatically tagged by the tagger trained on the training set.

4.1 Supertagging Experiments

We use the training data set to train a supertagger of each model using Conditional Random Fields (CRF) and the test data set to evaluate the accuracies. We use version 0.12 of CRFsuite[2] for our CRF implementation. First-order transitions, and word/POS of uni, bi and trigrams in a 7-word window surrounding the target word are used as features. Table 3 shows the result of the supertagging accuracies. The supertag accuracies are around 87-88% for both models, suggesting that most of the supertags can be effectively learned by standard CRFs. The tagger takes 0.001 and 0.005 second per sentence for Model 1 and 2 respectively.

In our error analysis, we find it is challenging to assign correct supertags for obligatory dependents of Model 2. In the test set, the number of the supertags encoding obligatory dependents is 5432 and its accuracy is 74.61% (The accuracy of the corresponding supertags in Model 1 is 82.18%). Among them, it is especially difficult to predict the supertags encoding obligatory dependents with a head information of subordination conjunction 'SBAR', such as 'SBAR/L+SUB/L_PRD/R'. The accuracy of such supertags is around 60% (e.g., the accuracy of a supertag 'SBAR/L+SUB/L_PRD/R' is 57.78%), while the supertags encoding dependents with a la-

[1] http://stp.lingfil.uu.se/ nivre/research/Penn2Malt.jar
[2] http://www.chokkan.org/software/crfsuite/

feature	Model1	Model2
baseline	90.25	90.25
+unigram of supertag	90.59	90.76
+bigram of supertag	91.37	91.08
+head-dependent	91.22	91.28

Table 4: Unlabeled attachment scores (UAS) on the development set for each feature template.

Model	UAS	Root	Complete
baseline	90.05	91.10	37.41
Model 1	91.35	93.17	41.35
Model 2	91.23	92.72	41.35

Table 5: Accuracies for English dependency parsing on the test set. UAS = unlabeled attachment score; Root = root attachment score; Complete = the percentage of sentences in which all tokens were assigned their correct heads.

bel 'VC' are assigned almost correctly (e.g., the accuracy of 'VC/L+VC/R' is 97.41%). A verb within a subordinating clause usually has the subordinating conjunction as its head and it tends to be long-range dependency, which is harder to predict. 'VC' represents verb complements. A gerund and a past participle is often a dependent of the immediate front verb, so it is not so difficult to identify the dependency relation.

4.2 Dependency Parsing Experiments

First, we evaluate the effectiveness of the feature templates proposed in Section 3. Following the same procedure as our POS tagger, we first assign supertags to the training data by ten-fold jackknifing, then train our Easy-First dependency parser on these predicted supertags. For development and test sets, we assign supertags based on a supertagger trained on the whole training data.

Table 4 shows the effect of new supertag features on the development data. We start with the baseline features, and incrementally add the unigrams, bigrams, and head-dependent feature templates. For Model 1 we observe that adding unigram features improve the baseline UAS slightly by 0.34% while additionally adding bigram features give larger improvements of 0.78%. On the other hand, for Model 2 unigram features make bigger contribution on improvements by 0.51% than bigram ones 0.32%. One possible explanation is that because each supertag of Model 2 encodes richer syntactic information, an individual tag can make bigger contribution on improvements than Model 1 as a unigram feature. However, since supertags of Model 2 can be erroneous and noisy combination of multiple supertags, such as bigram features, can propagate errors.

Using all features, the accuracy of the accuracy of Model 2 improved further by 0.20%, while Model 1 dropped by 0.15%. It is unclear why Model 1 accuracy dropped, but one hypothesis is that coarse-grained supertags may conflate some head-dependent. The development set UAS for combinations of all features are 91.22% (Model 1) and 91.28% (Model 2), corresponding to 0.97% and 1.03% improvement over the baseline.

Next, we show the parsing accuracies on the test set, using all unigram, bigram, and head-dependents supertag features. The UAS[3], Root attachment scores, and Complete accuracy are shown in Table 5. Both Model 1 and 2 outperform the baseline in all metrics. UAS improvements for both models are statistically significant under the McNemar test, $p < 0.05$ (difference between Model 1 and 2 is not significant). Notably, Model 1 achieves parsing improvements of 1.3% in unlabeled attachment, 2.07% root attachment, and 3.94% in complete accuracy. Comparing Model 1 to baseline, attachment improvements binned by distance to head are as follows: +0.54 F1 for distance 1, +0.81 for distance 2, +2.02 for distance 3 to 6, +2.95 for distance 7 or more, implying supertags are helpful for long distance dependencies.

5 Conclusions

We have demonstrated the effectiveness of supertags as features for English transition-based dependency parsing. In previous work, syntactic information, such as a head and dependents of a word, cannot be used as features before partial tree structures are constructed (Zhang and Nivre, 2011; Goldberg and Elhadad, 2010). By using supertags as features, we can utilize fine-grained syntactic information without waiting for partial trees to be built, and they contribute to improvement of accuracies of English dependency parsing. In future work, we would like to develop parsers that directly integrate supertag ambiguity in the parsing decision, and to investigate automatic pattern mining approaches to supertag design.

[3]For comparison, MaltParser and MSTParser with baseline features is 88.68% and 91.37% UAS respectively

References

Bharat R Ambati, Tejaswini Deoskar and Mark Steedman. 2013. Using CCG categories to improve Hindi dependency parsing. In *Proceedings of ACL*, pages 604-609, Sofia, Bulgaria, August.

Srinivas Bangalore and Aravind K. Joshi. 1999. Supertagging: An approach to almost parsing. *Compuational Linguistics*, 25(2):237-265.

Kilian Foth, Tomas By, and Wolfgang Menzel. 2006. Guiding a Constraint Dependency Parser with Supertags. In *Proceedings of COLING/ACL 2006*, pages 289-296, Sydney, Australia, July.

Yoav Goldberg and Michael Elhadad. 2010. An Efficient Algorithm for Easy-First Non-Directional Dependency Parsing. In *Proceedings of HLT/NAACL*, pages 742-750, Los Angeles, California, June.

Liang Huang and Kenji Sagae. 2010. Dynamic programming for linear-time incremental parsing. In *Proceedings of ACL*, pages 1077-1086, Uppsala, Sweden, July.

Mitchell. P. Marcus, Beatrice Santorini and Mary Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313-330

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryiğit, Sandra Kübler, Svetoslav, Marinov, and Erwin Marsi. 2007. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95-135

Joakim Nivre, Johan Hall, Jens Nilsson, Gülsen Eryiğit and Svetoslav, Marinov. 2006. Labeled pseudo-projective dependency parsing with support vector machines. In *Proceedings of CoNLL*, pages 221-225, New York, USA.

N Okazaki. 2007. CRFsuite: a fast implementation of Conditional Random Fields (CRFs). http://www.chokkan.org/software/crfsuite/.

H Yamada and Y Matsumoto. 2003. Statistical dependency analysis using support vector machines. In *Proceedings of IWPT*, Nancy, France.

Yue Zhang and Joakim Nivre. 2011. Transition-based Dependency Parsing with Rich Non-local Feaures. In *Proceedings of ACL*, pages 188-193, Porland, Oregon, June.

Yao-zhong Zhang, Takuya Matsuzaki and Jun'ichi Tsujii. 2010. A Simple Approach for HPSG Supertagging Using Dependency Information. In *Proceedings of HLT/NAACL*, pages 645-648, Los Angeles, California, June.

Improving Dependency Parsers using Combinatory Categorial Grammar

Bharat Ram Ambati **Tejaswini Deoskar** **Mark Steedman**
Institute for Language, Cognition and Computation
School of Informatics, University of Edinburgh
bharat.ambati@ed.ac.uk, {tdeoskar,steedman}@inf.ed.ac.uk

Abstract

Subcategorization information is a useful feature in dependency parsing. In this paper, we explore a method of incorporating this information via Combinatory Categorial Grammar (CCG) categories from a supertagger. We experiment with two popular dependency parsers (Malt and MST) for two languages: English and Hindi. For both languages, CCG categories improve the overall accuracy of both parsers by around 0.3-0.5% in all experiments. For both parsers, we see larger improvements specifically on dependencies at which they are known to be weak: long distance dependencies for Malt, and verbal arguments for MST. The result is particularly interesting in the case of the fast greedy parser (Malt), since improving its accuracy without significantly compromising speed is relevant for large scale applications such as parsing the web.

1 Introduction

Dependency parsers can recover much of the predicate-argument structure of a sentence, while being relatively efficient to train and extremely fast at parsing. Dependency parsers have been gaining in popularity in recent times due to the availability of large dependency treebanks for several languages and parsing shared tasks (Buchholz and Marsi, 2006; Nivre et al., 2007a; Bharati et al., 2012).

Ambati et al. (2013) showed that the performance of Malt (Nivre et al., 2007b) on the free word order language, Hindi, is improved by using lexical categories from Combinatory Categorial Grammar (CCG) (Steedman, 2000). In this paper, we extend this work and show that CCG categories are useful even in the case of English, a typologically different language, where parsing accuracy of dependency parsers is already extremely high. In addition, we also demonstrate the utility of CCG categories to MST (McDonald et al., 2005) for both languages. CCG lexical categories contain subcategorization information regarding the dependencies of predicates, including long-distance dependencies. We show that providing this subcategorization information in the form of CCG categories can help both Malt and MST on precisely those dependencies for which they are known to have weak rates of recovery. The result is particularly interesting for Malt, the fast greedy parser, as the improvement in Malt comes without significantly compromising its speed, so that it can be practically applied in web scale parsing. Our results apply both to English, a fixed word order and morphologically simple language, and to Hindi, a free word order and morphologically rich language, indicating that CCG categories from a supertagger are an easy and robust way of introducing lexicalized subcategorization information into dependency parsers.

2 Related Work

Parsers using different grammar formalisms have different strengths and weaknesses, and prior work has shown that information from one formalism can improve the performance of a parser in another formalism. Sagae et al. (2007) achieved a 1.4% improvement in accuracy over a state-of-the-art HPSG parser by using dependencies from a dependency parser for constraining wide-coverage rules in the HPSG parser. Coppola and Steedman (2013) incorporated higher-order dependency features into a cube decoding phrase-structure parser and obtained significant gains on dependency recovery for both in-domain and out-of-domain test sets.

Kim et al. (2012) improved a CCG parser using dependency features. They extracted n-best parses from a CCG parser and provided dependency

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 159–163,
Gothenburg, Sweden, April 26-30 2014.

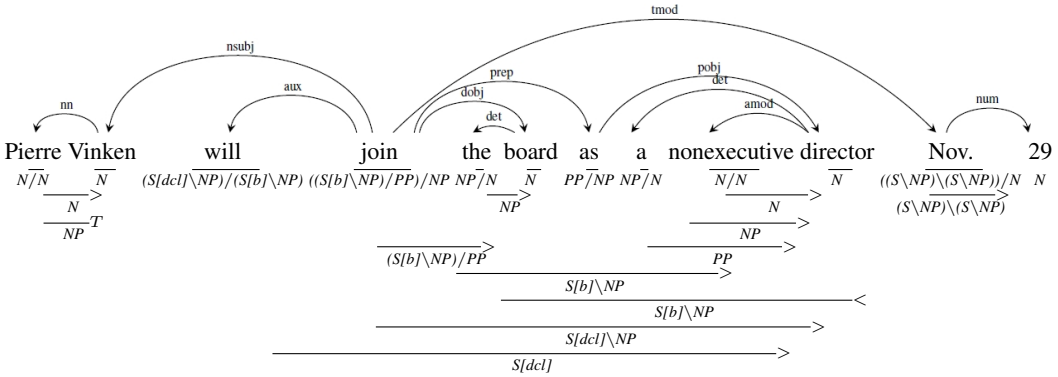


Figure 1: A CCG derivation and the Stanford scheme dependencies for an example sentence.

features from a dependency parser to a re-ranker with an improvement of 0.35% in labelled F-score of the CCGbank test set. Conversely, Ambati et al. (2013) showed that a Hindi dependency parser (Malt) could be improved by using CCG categories. Using an algorithm similar to Cakici (2005) and Uematsu et al. (2013), they first created a Hindi CCGbank from a Hindi dependency treebank and built a supertagger. They provided CCG categories from a supertagger as features to Malt and obtained overall improvements of 0.3% and 0.4% in unlabelled and labelled attachment scores respectively.

3 Data and Tools

Figure 1 shows a CCG derivation with CCG lexical categories for each word and Stanford scheme dependencies (De Marneffe et al., 2006) for an example English sentence. (Details of CCG and dependency parsing are given by Steedman (2000) and Kübler et al. (2009).)

3.1 Treebanks

In English dependency parsing literature, Stanford and CoNLL dependency schemes are widely popular. We used the Stanford parser's built-in converter (with the basic projective option) to generate Stanford dependencies and Penn2Malt[1] to generate CoNLL dependencies from Penn Treebank (Marcus et al., 1993). We used standard splits, training (sections 02-21), development (section 22) and testing (section 23) for our experiments. For Hindi, we worked with the Hindi Dependency Treebank (HDT) released as part of Coling 2012 Shared Task (Bharati et al., 2012). HDT contains 12,041 training, 1,233 development and 1,828 testing sentences.

We used the English (Hockenmaier and Steedman, 2007) and Hindi CCGbanks (Ambati et al., 2013) for our experiments. For Hindi we used two lexicons: a fine-grained one (with morphological information) and a coarse-grained one (without morphological information).

3.2 Supertaggers

We used Clark and Curran (2004)'s supertagger for English, and Ambati et al. (2013)'s supertagger for Hindi. Both are Maximum Entropy based CCG supertaggers. The Clark and Curran (2004) supertagger uses different features like word, part-of-speech, and contextual and complex bi-gram features to obtain a 1-best accuracy of 91.5% on the development set. In addition to the above mentioned features, Ambati et al. (2013) employed morphological features useful for Hindi. The 1-best accuracy of Hindi supertagger for fine-grained and coarse-grained lexicon is 82.92% and 84.40% respectively.

3.3 Dependency Parsers

There has been a significant amount of work on parsing English and Hindi using the Malt and MST parsers in the recent past (Nivre et al., 2007a; Bharati et al., 2012). We first run these parsers with previous best settings (McDonald et al., 2005; Zhang and Nivre, 2012; Bharati et al., 2012) and treat them as our baseline. In the case of English, Malt uses arc-standard and stack-projective parsing algorithms for CoNLL and Stanford schemes respectively and LIBLINEAR learner (Fan et al., 2008) for both the schemes. MST uses 1st-order features, and a projective parsing algorithm with 5-best MIRA training for both the schemes. For Hindi, Malt uses the arc-standard parsing algorithm with a LIBLINEAR learner. MST uses 2nd-order features, non-projective algorithm with 5-best MIRA training.

For English, we assigned POS-tags using a perceptron tagger (Collins, 2002). For Hindi, we also did all our experiments using automatic features

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

Language	*Experiment*	*Malt*		*MST*	
		UAS	*LAS*	*UAS*	*LAS*
English	Stanford Baseline	90.32	87.87	90.36	87.18
	Stanford + CCG	**90.56** (2.5)**	**88.16** (2.5)**	**90.93** (5.9)**	**87.73** (4.3)**
	CoNLL Baseline	89.99	88.73	90.94	89.69
	CoNLL + CCG	**90.38** (4.0)**	**89.19** (4.1)**	**91.48** (5.9)**	**90.23** (5.3)**
Hindi	Baseline	88.67	83.04	90.52	80.67
	Fine CCG	88.93** (2.2)	83.23* (1.1)	**90.97** (4.8)**	**80.94* (1.4)**
	Coarse CCG	**89.04** (3.3)**	**83.35* (1.9)**	90.88** (3.8)	80.73* (0.4)

Table 1: Impact of CCG categories from a supertagger on dependency parsing. Numbers in brackets are percentage of errors reduced. McNemar's test compared to baseline, * = $p < 0.05$; ** = $p < 0.01$ (Hindi Malt results (grey background) are from Ambati et al. (2013)).

(POS, chunk and morphological information) extracted using a Hindi shallow parser[2].

4 CCG Categories as Features

Following Ambati et al. (2013), we used supertags which occurred at least K times in the training data, and backed off to coarse POS-tags otherwise. For English K=1, i.e., when we use CCG categories for all words, gave the best results. K=15 gave the best results for Hindi due to sparsity issues, as the data for Hindi is small. We provided a supertag as an atomic symbol similar to a POS tag and didn't split it into a list of argument and result categories. We explored both Stanford and CoNLL schemes for English and fine and coarse-grained CCG categories for Hindi. All feature and parser tuning was done on the development data. We assigned automatic POS-tags and supertags to the training data.

4.1 Experiments with Supertagger output

We first used gold CCG categories extracted from each CCGbank as features to the Malt and MST, to get an upper bound on the utility of CCG categories. As expected, gold CCG categories boosted the Unlabelled Attachment Score (UAS) and Labelled Attachment Score (LAS) by a large amount (4-7% in all the cases).

We then experimented with using automatic CCG categories from the English and Hindi supertaggers as a feature to Malt and MST. With automatic categories from a supertagger, we got statistically significant improvements (McNemar's test, $p < 0.05$ for Hindi LAS and $p < 0.01$ for the rest) over the baseline parsers, for all cases (Table 1). Since the CCGbanks used to train the supertaggers are automatically generated from the constituency or dependency treebanks used to train the dependency parsers, the improvements are indeed due to reparameterization of the model to include CCG categories and not due to additional hand annotations in the CCGbanks. This shows that the rich subcategorization information provided by automatically assigned CCG categories can help Malt and MST in realistic applications.

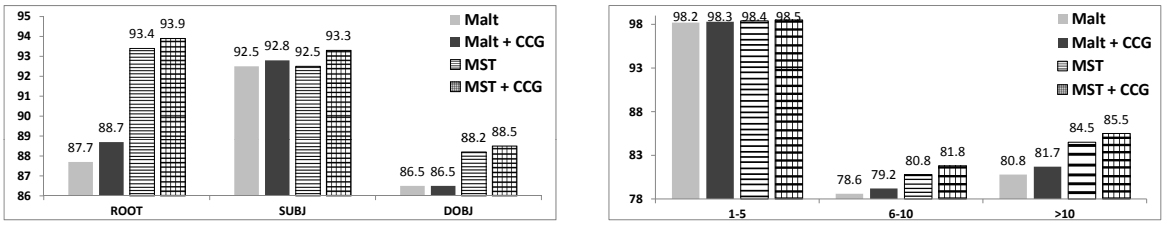
For English, in case of Malt, we achieved 0.3% improvement in both UAS and LAS for Stanford scheme. For CoNLL scheme, these improvements were 0.4% and 0.5% in UAS and LAS respectively. For MST, we got around 0.5% improvements in all cases.

In case of Hindi, fine-grained supertags gave larger improvements for MST. We got final improvements of 0.5% and 0.3% in UAS and LAS respectively. In contrast, for Malt, Ambati et al. (2013) had shown that coarse-grained supertags gave larger improvements of 0.3% and 0.4% in UAS and LAS respectively. Due to better handling of error propagation in MST, the richer information in fine-grained categories may have surpassed the slightly lower supertagger performance, compared to coarse-grained categories.

4.2 Analysis: English

We analyze the impact of CCG categories on different labels (label-wise) and distance ranges (distance-wise) for CoNLL scheme dependencies (We observed a similar impact for the Stanford scheme dependencies as well). Figure 2a shows the F-score for three major dependency labels, namely, `ROOT` (sentence root), `SUBJ` (subject), `OBJ` (object). For Malt, providing CCG categories gave an increment of 1.0%, 0.3% for `ROOT` and `SUBJ` labels respectively. For MST, the improvements for `ROOT` and `SUBJ` were 0.5% and 0.8% respectively. There was no significant improvement for `OBJ` label, especially in the case of Malt.

[2]http://ltrc.iiit.ac.in/analyzer/hindi/



(a) Label-wise impact

(b) Distance-wise impact

Figure 2: Label-wise and Distance-wise impact of supertag features on Malt and MST for English

Figure 2b shows the F-score of dependencies based on the distance ranges between words. The percentage of dependencies in the $1-5$, $6-10$ and >10 distance ranges are 88.5%, 6.6% and 4.9% respectively out of the total of around 50,000 dependencies. For both Malt and MST, there was very slight improvement for short distance dependencies ($1-5$) but significant improvements for longer distances ($6-10$ and >10). For Malt, there was an improvement of 0.6% and 0.9% for distances $6-10$, and >10 respectively. For MST, these improvements were 1.0% and 1.0% respectively.

4.3 Analysis: Hindi

In the case of Hindi, for MST, providing CCG categories gave an increment of 0.5%, 0.4% and 0.3% for `ROOT`, `SUBJ` and `OBJ` labels respectively in F-score over the baseline. Ambati et al. (2013) showed that for Hindi, providing CCG categories as features improved Malt in better handling of long distance dependencies.

The percentage of dependencies in the $1-5$, $6-10$ and >10 distance ranges are 82.2%, 8.6% and 9.2% respectively out of the total of around 40,000 dependencies. Similar to English, there was very slight improvement for short distance dependencies ($1-5$). But for longer distances, $6-10$, and >10, there was significant improvement of 1.3% and 1.3% respectively for MST. Ambati et al. (2013) reported similar improvements for Malt as well.

4.4 Discussion

Though valency is a useful feature in dependency parsing (Zhang and Nivre, 2011), Zhang and Nivre (2012) showed that providing valency information dynamically, in the form of the number of dependencies established in a particular state during parsing, did not help Malt. However, as we have shown above, providing this information as a static lexical feature in the form of CCG categories does help Malt. In addition to specifying the number of arguments, CCG categories also contain syntactic type and direction of those arguments. However, providing CCG categories as features to zpar (Zhang and Nivre, 2011) didn't have significant impact as it is already using similar information.

4.5 Impact on Web Scale Parsing

Greedy parsers such as Malt are very fast and are practically useful in large-scale applications such as parsing the web. Table 2, shows the speed of Malt, MST and zpar on parsing English test data in CoNLL scheme (including POS-tagging and supertagging time). Malt parses 310 sentences per second, compared to 35 and 11 of zpar and MST respectively. Clearly, Malt is orders of magnitude faster than MST and zpar. After using CCG categories from the supertagger, Malt parses 245 sentences per second, still much higher than other parsers. Thus we have shown a way to improve Malt without significantly compromising speed, potentially enhancing its usefulness for web scale parsing.

Parser	*Ave. Sents / Sec*	*Total Time*
MST	11	3m 36s
zpar	35	1m 11s
Malt	310	0m 7.7s
Malt + CCG	245	0m 10.2s

Table 2: Time taken to parse English test data.

5 Conclusion

We have shown that informative CCG categories, which contain both local subcategorization information and capture long distance dependencies elegantly, improve the performance of two dependency parsers, Malt and MST, by helping in recovering long distance relations for Malt and local verbal arguments for MST. This is true both in the case of English (a fixed word order language) and Hindi (free word order and morphologically richer language), extending the result of Ambati et al. (2013). The result is particularly interesting in the case of Malt which cannot directly use valency information, which CCG categories provide indirectly. It leads to an improvement in performance without significantly compromising speed and hence promises to be applicable to web scale processing.

References

Bharat Ram Ambati, Tejaswini Deoskar, and Mark Steedman. 2013. Using CCG categories to improve Hindi dependency parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 604–609, Sofia, Bulgaria.

Akshar Bharati, Prashanth Mannem, and Dipti Misra Sharma. 2012. Hindi Parsing Shared Task. In *Proceedings of Coling Workshop on Machine Translation and Parsing in Indian Languages*, Kharagpur, India.

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

Ruken Cakici. 2005. Automatic induction of a CCG grammar for Turkish. In *Proceedings of the ACL Student Research Workshop*, pages 73–78, Ann Arbor, Michigan.

Stephen Clark and James R. Curran. 2004. The importance of supertagging for wide-coverage CCG parsing. In *Proceedings of COLING-04*, pages 282–288.

Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *Proceedings of the conference on Empirical methods in natural language processing*, EMNLP '02, pages 1–8.

Greg Coppola and Mark Steedman. 2013. The effect of higher-order dependency features in discriminative phrase-structure parsing. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 610–616, Sofia, Bulgaria.

Marie Catherine De Marneffe, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *In LREC 2006*.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, June.

Julia Hockenmaier and Mark Steedman. 2007. CCGbank: A Corpus of CCG Derivations and Dependency Structures Extracted from the Penn Treebank. *Computational Linguistics*, 33(3):355–396.

Sunghwan Mac Kim, Dominick Ng, Mark Johnson, and James Curran. 2012. Improving combinatory categorial grammar parse reranking with dependency grammar features. In *Proceedings of COLING 2012*, pages 1441–1458, Mumbai, India.

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

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Online large-margin training of dependency parsers. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 91–98, Ann Arbor, Michigan.

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

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007b. Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Kenji Sagae, Yusuke Miyao, and Jun'ichi Tsujii. 2007. HPSG parsing with shallow dependency constraints. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 624–631, Prague, Czech Republic.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA, USA.

Sumire Uematsu, Takuya Matsuzaki, Hiroki Hanaoka, Yusuke Miyao, and Hideki Mima. 2013. Integrating multiple dependency corpora for inducing wide-coverage Japanese CCG resources. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1042–1051, Sofia, Bulgaria.

Yue Zhang and Joakim Nivre. 2011. Transition-based dependency parsing with rich non-local features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 188–193, Portland, Oregon, USA.

Yue Zhang and Joakim Nivre. 2012. Analyzing the effect of global learning and beam-search on transition-based dependency parsing. In *Proceedings of COLING 2012: Posters*, pages 1391–1400, Mumbai, India, December.

Fast and Accurate Unlexicalized Parsing via Structural Annotations

Maximilian Schlund, Michael Luttenberger, and Javier Esparza
Institut für Informatik
Technische Universität München
Boltzmannstraße 3
D-85748 Garching
{schlund,luttenbe,esparza}@model.in.tum.de

Abstract

We suggest a new annotation scheme for unlexicalized PCFGs that is inspired by formal language theory and only depends on the structure of the parse trees. We evaluate this scheme on the TüBa-D/Z treebank w.r.t. several metrics and show that it improves both parsing accuracy and parsing speed considerably. We also show that our strategy can be fruitfully combined with known ones like parent annotation to achieve accuracies of over 90% labeled F_1 and leaf-ancestor score. Despite increasing the size of the grammar, our annotation allows for parsing more than twice as fast as the PCFG baseline.

1 Introduction

As shown by (Klein and Manning, 2003), unlexicalized PCFGs can achieve high parsing accuracies when training trees are annotated with additional information. An annotation basically amounts to splitting each nonterminal into several subcategories, which can even be derived automatically (Petrov et al., 2006; Petrov and Klein, 2007). Currently used annotation strategies, e.g. parent annotation (Johnson, 1998) or selectively splitting special nonterminals (e.g. marking relative clauses) as in (Schiehlen, 2004), are mostly linguistically motivated (with the exception of the above mentioned automatic approach).

In this paper we study new heuristics motivated by formal language theory for improving the parsing accuracy of unlexicalized PCFGs by means of refining the nonterminals of the grammar: One heuristic splits a nonterminal X into a family of nonterminals $(X_d)_{d\in D}$ based on the notion of the *dimension* (also *Horton-Strahler number*) of a tree (Strahler, 1952; Esparza et al., 2007; Esparza et al., 2014).

The *dimension* of a rooted tree t is defined as the height of the highest perfect binary tree[1] we can obtain from t by pruning subtrees and contracting edges.[2]

A result of (Flajolet et al., 1979) shows that the dimension characterizes the *minimal* amount of memory that is required to traverse a tree. So, intuitively, parse trees of high dimension should indicate an unnaturally complex sentence structure requiring the reader to remember too many incomplete dependent clauses in the course of reading the sentence. Section 2 corroborates experimentally that, indeed, parse trees of natural language have small dimension.

Since dimension is a meaningful measure of complexity and parse trees have low dimension, we conjectured that annotating nonterminals with the dimension of the subtree rooted at them could improve parsing accuracy (see Fig. 1 for an illustration). Section 5 shows that this is indeed the case: The combination of the dimension annotation and the well known parent annotation technique leads to absolute improvements of more than 5% F_1, 7–8% leaf-ancestor score, and a relative reduction of the number of crossing brackets of over 25% compared to a plain PCFG baseline. At the same time, quite surprisingly, parsing speed more than doubles.

It could be argued that any other graph theoretical measure for the complexity of a tree could lead to similar results. For this reason we have also considered annotating nonterminals with the height of the subtree rooted at them (the height is the most basic measure related to trees). Our experiments show that height annotation is also beneficial but further refinement via parent annotation yields less improvements than for the dimension annotation.

[1] A binary tree of height h is *perfect* if it has 2^h leaves.

[2] In other words, the dimension of t is the height of the highest perfect binary tree which is a minor of t.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 164–168,
Gothenburg, Sweden, April 26-30 2014.

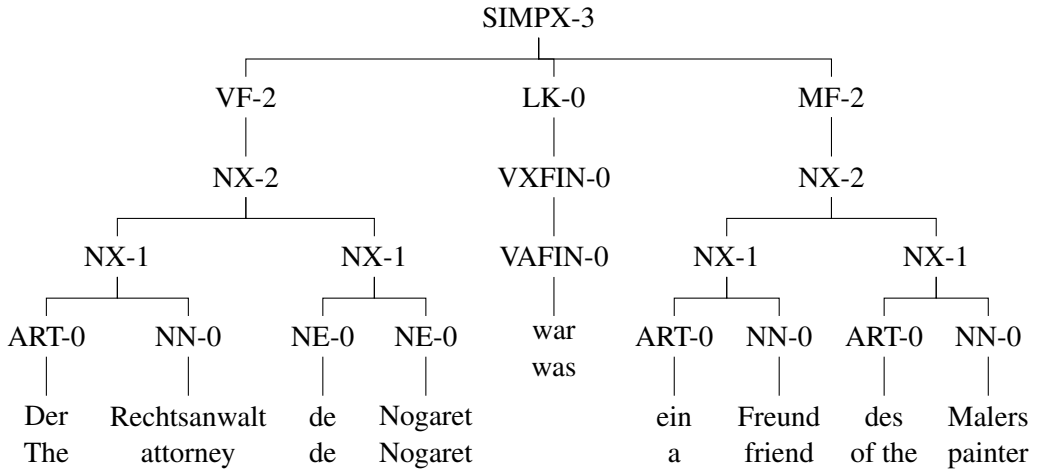


Figure 1: Dimension annotation of a tree from TüBa-D/Z: the label of every nonterminal is decorated with the dimension of the subtree rooted at it. The dimension of a parent node is the maximum of the dimensions of its children (*plus one* if this maximum is attained at least twice).

In the following two sections, we present more details on the use of tree dimension in NLP, continue with describing our experiments (Section 4) together with their results (Section 5), and finally conclude with some ideas for further improvements.

2 Tree Dimension of Natural Languages

We were able to validate our conjecture that parse trees of natural language should typically have small dimension on several treebanks for a variety of languages (cf. Table 1). The average dimension of parse trees varies only from 1.7 to 2.4 over all languages and the maximum dimension we ever encountered is 4.

Language	Average	Maximum
Basque	2.12	3
English	2.38	4
French	2.29	4
German(1)	1.94	4
German(2)	2.13	4
Hebrew	2.44	4
Hungarian	2.11	4
Korean	2.18	4
Polish	1.68	3
Swedish	1.83	4

Table 1: Average and maximum dimension for several treebanks of natural languages. Sources: English – 10% sample from the Penn treebank shipped with python nltk (Loper and Bird, 2002), German(2) – release 8 of the TüBa-D/Z treebank (Telljohann et al., 2003), the remaining treebanks are taken from the SPMRL shared task dataset (Seddah et al., 2013).

3 Annotation Methods

In this paper we compare three different annotation methods: dimension, height, and parent annotation. The dimension (resp. height) annotation refine a given nonterminal X by *annotating* it with the dimension (resp. height) of the subtree rooted at it. A standard technique in unlexicalized parsing we compare against is *vertical markovization*, i.e. to refine nonterminals by annotating them with their parent (or grandparent) nonterminal (Klein and Manning, 2003).

Let us remark that we focus here only on methods to split nonterminals and leave merging strategies for further investigations. Amongst them *horizontal markovization* (Klein and Manning, 2003) is especially valuable for battling sparsity and can lead to more compact and often more accurate PCFGs.

4 Methodology

4.1 Experimental Setup

We use release 8 of the TüBa-D/Z treebank (Telljohann et al., 2003) as dataset. To combine easy prototyping and data exploration with efficient parsing and standard evaluation methods we used python nltk (Loper and Bird, 2002) together with the Stanford parser (Klein and Man-

ning, 2003). For evaluation we used the built in evalb, leaf-ancestor, and crossing brackets metrics provided by the Stanford parser. Is is important to note that all our experiments use gold tags from the treebank[3] which had the pleasant side effect that no parse failures were encountered. All experiments were carried out on a machine with an Intel i7 2.7 GHz CPU and 8 GB RAM and took about one week to run[4]. Our scripts and raw data can be obtained freely from `https://github.com/mschlund/nlp-newton`.

4.2 Randomization

We decided to sample our training- and test-data randomly from the treebank several times independently for each annotation strategy under test. This enables us to give more precise estimations of parsing accuracy (Section 5) and to assess their variability (cf. Figure 2). For each sample size N from $\{5k, 10k, 20k, \ldots, 70k\}$ we selected a random sample of size N from the set of all 75408 trees in the treebank. The first 90% of this sample was used as training set and the remaining 10% as test set. We then evaluated each of our six annotation methods on this same training/test set. The whole process was repeated ten times each, yielding 480 experiments altogether. For each experiment we evaluated parsing accuracy according to three evaluation measures as well as the parsing speed and the size of the derived grammar. Each of these numbers was then averaged over the ten random trials. To ensure perfect reproducibility we saved the seeds we used to seed the random generator.

4.3 Evaluation Measures

To thoroughly assess the performance of our annotation schemes we not only report the usual constituency measures (labeled precision/recall/F_1 and crossing brackets) proposed originally by (Abney et al., 1991) but also calculate leaf-ancestor scores (LA) proposed by (Sampson, 2000) since it has been argued that LA-scores describe the informal notion of a "good" parse better than the usual constituency measures. This is especially relevant for comparing parsing accuracy over different treebanks (Rehbein and Van Genabith, 2007a; Rehbein and van Genabith, 2007b).

[3]This is unrealistic of course, but is used for comparability with other work like (Rafferty and Manning, 2008).

[4]We only used a single core, since memory turned out to be the main bottleneck.

5 Results

Our results are collected in Table 5. We measured a baseline accuracy of 84.8% labeled F_1-score for a plain PCFG without any annotations, lower than the 88% reported by (Rafferty and Manning, 2008) on a previous release of the TüBa-D/Z treebank (comprising only $20k$ sentences of length at most 40). However, the absolute improvements we found using annotations are consistent with their work, e.g. our experiments show an absolute increase of 3.4% when using parent annotation while (Rafferty and Manning, 2008) report a 3.1% increase. We suspect that the differences are largely suspect to the different data: considering sentences up to length 40, our experiments yield scores that are 1% higher. To explain all remaining differences we plan to replicate their setup.

5.1 Impact of Annotations

All three annotation methods (w.r.t. parent, dimension, height which we will abbreviate by PA, DA, HA for convenience) lead to comparable improvements w.r.t. constituency measures with small advantages for the two structural annotations. LA-evaluation on the other hand shows that HA and DA have a clear advantage of 3% over PA.

Quite surprisingly, both DA and HA can be fruitfully combined with parent annotation improving F_1 further by almost 2% and LA-metrics by 1–2% as well. However, the height+parent combination cannot compete with the dimension+parent method. One reason for this might be the significant increase in grammar size and resulting data-sparseness problems, although our learning curves (cf. Figure 2) suggest that lack of training data is not an issue.

Altogether, the DA+PA combination is the most precise one w.r.t. all metrics. It provides absolute increases of 5.6% labeled F_1 and 7.4–8.4% LA-score and offers a relative reduction of crossing brackets by 27%. This is especially relevant since according to (Manning and Schütze, 1999) a high number of crossing brackets is often considered "particularly dire". Finally, this combination leads to a 60% increase in the number of exactly parsed sentences, significantly more than for the other methods.

5.2 Parsing Speed

We further study to what extent the three heuristics increase the size of the grammar and the time

Annotation	$\|G\|$	Speed $\pm$ stderr	evalb F_1	evalb exact	Leaf-Ancestor LA (s)	Leaf-Ancestor LA (c)	Crossing brackets # CB	Crossing brackets zero CB
Plain	21009	1.74 ± 0.04	84.8	24.4	84.0	79.7	1.17	58.5
Parent	34192	1.07 ± 0.01	88.2	31.8	86.6	82.9	1.07	61.8
Height	76096	3.06 ± 0.03	88.7	33.7	89.8	86.2	0.93	65.2
Height+parent	130827	2.20 ± 0.04	89.2	36.8	90.8	87.0	0.95	65.4
Dim	49798	$\mathbf{6.02} \pm 0.10$	88.5	31.8	89.7	86.1	0.90	64.9
Dim+parent	84947	4.04 ± 0.07	**90.4**	**39.1**	**91.4**	**88.1**	**0.85**	**67.2**

Table 2: Average grammar sizes, parsing speed, and parsing accuracies according to various metrics (for the $70k$ samples only, i.e. on 7000 test trees). All numbers are averaged over 10 independent random samples. $|G|$ denotes the number of rules in the grammar, parsing speed is measured in sentences per second. LA scores are reported as sentence-level (s) and corpus-level (c) averages, respectively. All accuracies reported in % (except # CB – the average number of crossing brackets per sentence).

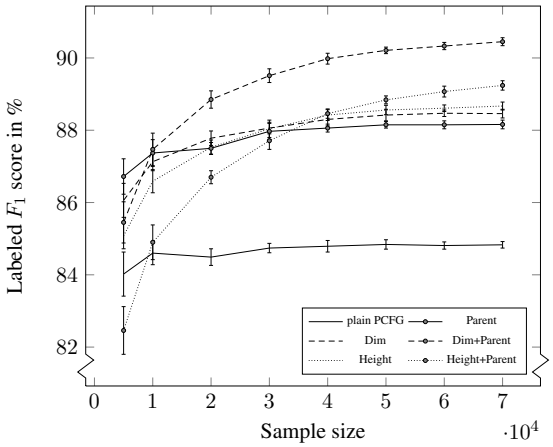


Figure 2: Learning curves for different annotation strategies. Average F_1 with standard deviation for random samples of various sizes (10 independent samples each).

needed to parse a sentence. As expected all three annotations increase the size of the grammar considerably (PA by 60%, DA by almost 140%, and HA by 260%). Surprisingly, our experiments did not show a direct influence of the grammar size on the average time needed to parse a tree: While parsing speed for PA drops by about 40%, DA and HA actually lead to significant *speedups* over the baseline (factor 3.4 for DA and 1.7 for HA). For the combination of dimension and parent annotation the gain in speed is less pronounced but still a factor of 2.3. One possible explanation is the fact that (for a grammar in CNF) a nonterminal of dimension d can only be produced either by combining one of dimension d with one of dimension strictly less than d or by two of dimension exactly $d-1$. Since the dimensions involved are typically very small (cf. Table 1) this may restrict the search space significantly.

6 Discussion

We have described a new and simple yet effective annotation strategy to split nonterminals based on the purely graph-theoretic concept of *tree dimension*. We show that annotating nonterminals with either their dimension or their height gives accuracies that lie beyond parent annotation. Furthermore dimension and parent annotation in combination yield even higher accuracies (90.4% labeled F_1 and 91.4% LA-score on a sentence-level). Lastly, one of the most surprising findings is that, despite considerable growth of grammar size, parsing is significantly faster.

6.1 Future Work

We are currently experimenting with other treebanks like the SPMRL dataset (Seddah et al., 2013) which contains various "morphologically rich" languages (cf. Table 1). Although we cannot possibly expect to match the accuracies achieved by highly optimized lexicalized parsers with our simple annotation strategy alone, we are confident that our results transfer to other languages. A logical next step is to integrate our annotation methods into current parsing frameworks.

Since our annotations increase the size of the grammar significantly, horizontal markovization and more careful, selective dimension/height-splits (i.e. only carry out "profitable" splits) seem promising to avoid problems of data-sparsity – in particular if one wants to use further state-splitting techniques that are more linguistically motivated.

Finally, we are interested in understanding the parsing speedup incurred by dimension/height-annotations and to provide a theoretical analysis.

References

S. Abney, S. Flickenger, C. Gdaniec, C. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini, and T. Strzalkowski. 1991. Procedure for Quantitatively Comparing the Syntactic Coverage of English Grammars. In E. Black, editor, *Proceedings of the Workshop on Speech and Natural Language*, HLT '91, pages 306–311, Stroudsburg, PA, USA. Association for Computational Linguistics.

Javier Esparza, Stefan Kiefer, and Michael Luttenberger. 2007. An Extension of Newton's Method to ω-Continuous Semirings. In *Developments in Language Theory*, volume 4588 of *LNCS*, pages 157–168. Springer.

Javier Esparza, Michael Luttenberger, and Maximilian Schlund. 2014. A Brief History of Strahler Numbers. In *Language and Automata Theory and Applications*, volume 8370 of *Lecture Notes in Computer Science*, pages 1–13. Springer International Publishing.

Philippe Flajolet, Jean-Claude Raoult, and Jean Vuillemin. 1979. The Number of Registers Required for Evaluating Arithmetic Expressions. *Theoretical Computer Science*, 9:99–125.

Mark Johnson. 1998. PCFG Models of Linguistic Tree Representations. *Computational Linguistics*, 24(4):613–632.

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

Edward Loper and Steven Bird. 2002. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, pages 63–70. Association for Computational Linguistics.

Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of statistical natural language processing*, volume 999. MIT Press.

Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. In *HLT-NAACL*, pages 404–411.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 433–440. Association for Computational Linguistics.

Anna N. Rafferty and Christopher D. Manning. 2008. Parsing Three German Treebanks: Lexicalized and Unlexicalized Baselines. In *Proceedings of the Workshop on Parsing German*, PaGe '08, pages 40–46, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ines Rehbein and Josef Van Genabith. 2007a. Evaluating Evaluation Measures. In *NODALIDA*, pages 372–379.

Ines Rehbein and Josef van Genabith. 2007b. Treebank Annotation Schemes and Parser Evaluation for German. In *EMNLP-CoNLL*, pages 630–639.

Geoffrey Sampson. 2000. A Proposal for Improving the Measurement of Parse Accuracy. *International Journal of Corpus Linguistics*, 5(1):53–68.

Michael Schiehlen. 2004. Annotation strategies for probabilistic parsing in german. In *Proceedings of the 20th international conference on Computational Linguistics*, COLING '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

Djamé Seddah, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho D. Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiorkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska, and Eric Villemonte de la Clérgerie. 2013. Overview of the SPMRL 2013 Shared Task: A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In *Proceedings of the 4th Workshop on Statistical Parsing of Morphologically Rich Languages: Shared Task*, Seattle, WA.

Arthur N. Strahler. 1952. Hypsometric (Area-Altitude) Analysis of Erosional Topology. *Bulletin of the Geological Society of America*, 63(11):1117–1142.

Heike Telljohann, Erhard W. Hinrichs, Sandra Kübler, Heike Zinsmeister, and Kathrin Beck. 2003. Stylebook for the Tübingen Treebank of Written German (TüBa-D/Z). Seminar für Sprachwissenschaft, Universität Tübingen, Germany.

Data Driven Language Transfer Hypotheses

Ben Swanson
Brown University
Providence, RI
chonger@cs.brown.edu

Eugene Charniak
Brown University
Providence, RI
ec@cs.brown.edu

Abstract

Language transfer, the preferential second language behavior caused by similarities to the speaker's native language, requires considerable expertise to be detected by humans alone. Our goal in this work is to replace expert intervention by data-driven methods wherever possible. We define a computational methodology that produces a concise list of lexicalized syntactic patterns that are controlled for redundancy and ranked by relevancy to language transfer. We demonstrate the ability of our methodology to detect hundreds of such candidate patterns from currently available data sources, and validate the quality of the proposed patterns through classification experiments.

1 Introduction

The fact that students with different native language backgrounds express themselves differently in second language writing samples has been established experimentally many times over (Tetreault et al., 2013), and is intuitive to most people with experience learning a new language. The exposure and understanding of this process could potentially enable the creation of second language (L2) instruction that is tailored to the native language (L1) of students.

The detectable connection between L1 and L2 text comes from a range of sources. On one end of the spectrum are factors such as geographic or cultural preference in word choice, which are a powerful L1 indicator. On the other end lie linguistic phenomena such as language transfer, in which the preferential over-use or under-use of structures in the L1 is reflected in the use of corresponding patterns in the L2. We focus on language transfer in this work, based on our opinion that such effects are more deeply connected to and effectively utilized in language education.

The inherent challenge is that viable language transfer hypotheses are naturally difficult to construct. By the requirement of contrasting different L1 groups, hypothesis formulation requires deep knowledge of multiple languages, an ability reserved primarily for highly trained academic linguists. Furthermore, the sparsity of any particular language pattern in a large corpus makes it difficult even for a capable multilingual scholar to detect the few patterns that evidence language transfer. This motivates data driven methods for hypothesis formulation.

We approach this as a representational problem, requiring the careful definition of a class of linguistic features whose usage frequency can be determined for each L1 background in both L1 and L2 text (e.g. both German and English written by Germans). We claim that a feature exhibiting a sufficiently non-uniform usage histogram in L1 that is mirrored in L2 data is a strong language transfer candidate, and provide a quantified measure of this property.

We represent both L1 and L2 sentences in a universal constituent-style syntactic format and model language transfer hypotheses with contiguous syntax sub-structures commonly known as Tree Substitution Grammar (TSG) fragments (Post and Gildea, 2009)(Cohn and Blunsom, 2010). With these features we produce a concise ranked list of candidate language transfer hypotheses and their usage statistics that can be automatically augmented as increasing amounts of data become available.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 169–173,
Gothenburg, Sweden, April 26-30 2014.

2 Related Work

This work leverages several recently released data sets and analysis techniques, with the primary contribution being the transformations necessary to combine these disparate efforts. Our analysis methods are closely tied to those described in Swanson and Charniak (2013), which contrasts techniques for the discovery of discriminative TSG fragments in L2 text. We modify and extend these methods to apply to the universal dependency treebanks of McDonald et al. (2013), which we will refer to below to as the UTB. Bilingual lexicon construction (Haghighi et al., 2008) is also a key component, although previous work has focused primarily on nouns while we focus on stopwords. We also transform the UTB into constituent format, in a manner inspired by Carroll and Charniak (1992).

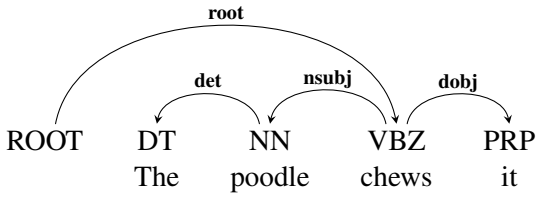
There is a large amount of related research in Native Language Identification (NLI), the task of predicting L1 given L2 text. This work has culminated in a well attended shared task (Tetreault et al., 2013), whose cited report contains an excellent survey of the history of this task. In NLI, however, L1 data is not traditionally used, and patterns are learned directly from L2 text that has been annotated with L1 labels. One notable outlier is Brooke and Hirst (2012), which attempts NLI using only L1 data for training using large online dictionaries to tie L2 English bigrams and collocations to possible direct translations from native languages. Jarvis and Crossley (2012) presents another set of studies that use NLI as a method to form language transfer hypotheses.

3 Methodology

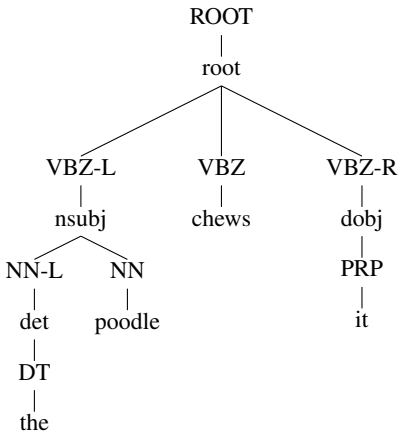
The first of the four basic requirements of our proposed method is the definition of a class of features $\mathcal{F}$ such that a single feature $F \in \mathcal{F}$ is capable of capturing language transfer phenomenon. The second is a universal representation of both L1 and L2 data that allows us to count the occurrences of any F in an arbitrary sentence. Third, as any sufficiently expressive $\mathcal{F}$ is likely to be very large, a method is required to propose an initial candidate list $C \subset \mathcal{F}$. Finally, we refine C into a ranked list H of language transfer hypotheses, where H has also been filtered to remove redundancy.

In this work we define $\mathcal{F}$ to be the set of Tree Substitution Grammar (TSG) fragments in our data, which allows any connected syntactic structure to be used as a feature. As such, our universal representation of L1/L2 data must be a constituent tree structure of the general form used in syntactic parsing experiments on the Penn Treebank. The UTB gets us most of the way to our goal, defining a dependency grammar with a universal set of part of speech (POS) tags and dependency arc labels.

Two barriers remain to the use of standard TSG induction algorithms. The first is to define a mapping from the dependency tree format to constituency format. We use the following dependency tree to illustrate our transformation.



Under our transformation, the above dependency parse becomes



We also require a multilingual lexicon in the form of a function $M_L(w)$ for each language L that maps words to clusters representing their meaning. In order to avoid cultural cues and reduce noise in our mapping, we restrict ourselves to clusters that correspond to a list of L2 stopwords. Any L2 words that do not appear on this list are mapped to the unknown "UNK" symbol, as are all foreign words that are not good translations of any L2 stopword. Multiple words from a single language can map to the same cluster, and it is worth noting that this is true for L2 stopwords as well.

To determine the mapping functions M_L we train IBM translation models in both directions between the L2 and each L1. We create a graph in which nodes are words, either the L2 stopwords or any L1 word with some translation probability to

or from one of the L2 stopwords. The edges in this graph exist only between L2 and L1 words, and are directed with weight equal to the IBM model's translation probability of the edge's target given its source. We construct M_L by removing edges with weight below some threshold and calculating the connected components of the resulting graph. We then discard any cluster that does not contain at least one word from each L1 and at least one L2 stopword.

To propose a candidate list C, we use the TSG induction technique described in Swanson and Charniak (2013), which simultaneously induces multiple TSGs from data that has been partitioned into labeled types. This method permits linguistically motivated constraints as to which grammars produce each type of data. For an experimental setup that considers n different L1s, we use $2n+1$ data types; Figure 1 shows the exact layout used in our experiments. Besides the necessary n data types for each L1 in its actual native language form and n in L2 form, we also include L2 data from L2 native speakers. We also define $2n + 1$ grammars. We begin with n grammars that can each be used exclusively by one native language data type, representing behavior that is unique to each native language (grammars A-C in Figure 1) . This is done for the L2 as well (grammar G). Finally, we create an interlanguage grammar for each of our L1 types that can be used in derivation of both L1 and L2 data produced by speakers of that L1 (grammars D-F).

The final step is to filter and rank the TSG fragments produced in C, where filtering removes redundant features and ranking provides some quantification of our confidence in a feature as a language transfer hypothesis. Swanson and Charniak (2013) provides a similar method for pure L2 data, which we modify for our purposes. For redundancy filtering no change is necessary, and we use their recommended Symmetric Uncertainty method. For a ranking metric of how well a fragment fits the profile of language transfer we adopt the expected per feature loss (or risk) also described in their work. For an arbitrary feature F, this is defined as

$$\mathcal{R}(F) = \frac{1}{|T_F|} \sum_{t \in T_F} P_F(L \neq L_t^*)$$

where T_F is the subset of the test data that contains the feature F, and L_t^* is the gold label of test datum t. While in their work the predictive distribution $P_F(L)$ is determined by the observed counts of F in L2 training data, we take our estimates directly from the L1 data of the languages under study. This metric captures the extent to which the knowledge of a feature F's L1 usage can be used to predict its usage in L2.

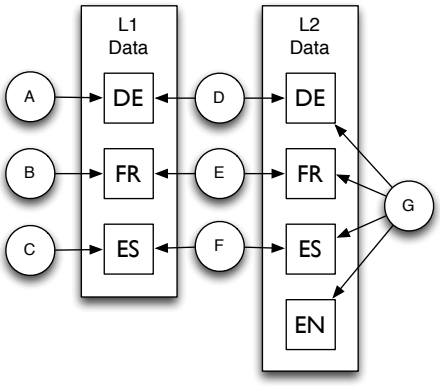


Figure 1: The multi-grammar induction setup used in our experiments. Squares indicate data types, and circles indicate grammars. Data type labels indicate the native language of the speaker, and all L2 data is in English.

The final result is a ranked and filtered list of hypotheses H. The elements of H can be subjected to further investigation by experts and the accompanying histogram of counts contains the relevant empirical evidence. As more data is added, the uncertainty in the relative proportions of these histograms and their corresponding $\mathcal{R}$ is decreased. One additional benefit of our method is that TSG induction is a random process, and repeated runs of the sampling algorithm can produce different features. Since redundancy is filtered automatically, these different feature lists can be combined and processed to potentially find additional features given more computing time.

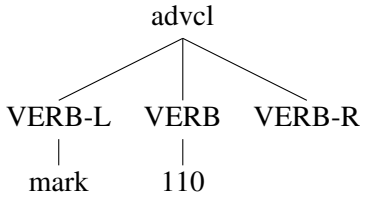
4 Results

Limited by the intersection of languages across data sets, we take French, Spanish, and German as our set of L1s with English as the L2. We use the UTB for our native language data, which provides around 4000 sentences of human annotated text for each L1. For our L2 data we use the ETS Corpus of Non-Native English (Blanchard et al., 2013), which consists of over 10K sentences per L1 label drawn from TOEFL® exam essays. Fi-

nally, we use the Penn Treebank as our source of native English data, for a total of seven data types; four in English, and one in each L1.

When calculating metrics such as redundancy and $\mathcal{R}(F)$ we use all available data. For TSG sampling, we balance our data sets to 4000 sentences from each data type and sample using the Enbuske sampler that was released with Swanson and Charniak (2013). To construct word clusters, we use Giza++ (Och and Ney, 2003) and train on the Europarl data set (Koehn, 2005), using .25 as a threshold for construction on connected components.

We encourage the reader to peruse the full list of results[1], in which each item contains the information in the following example.



	ES	DE	FR
L1	4.2	0.0	0.0
L2	2.3	0.3	0.3

This fragment corresponds to an adverbial clause whose head is a verb in the cluster 110, which contains the English word "is" and its various translations. This verb has a single left dependent, a clause marker such as "because", and at least one right dependent. Its prevalence in Spanish can explained by examining the translations of the English sentence "I like it because it is red".

ES Me gusta porque es rojo.
DE Ich mag es, weil es rot ist.
FR Je l'aime parce qu'il est rouge.

Only in the Spanish sentence is the last pronoun dropped, as in "I like it because is red". This observation, along with the L1/L2 profile which shows the count per thousand sentences in each language provides a strong argument that this pattern is indeed a form of language transfer.

Given our setup of three native languages, a feature with $\mathcal{R}(F) < .66$ is a candidate for language transfer. However, several members of our filtered list have $\mathcal{R}(F) > .66$, which is to say that their

[1] bllip.cs.brown.edu/download/interlanguage_corpus.pdf

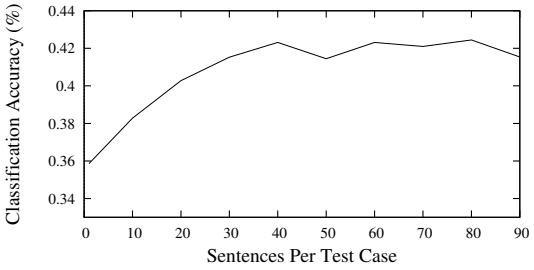


Figure 2: Creating test cases that consist of several sentences mediates feature sparsity, providing clear evidence for the discriminative power of the chosen feature set.

L2 usage does not mirror L1 usage. This is to be expected in some cases due to noise, but it raises the concern that our features with $\mathcal{R}(F) < .66$ are also the result of noise in the data. To address this, we apply our features to the task of cross language NLI using only L1 data for training. If the variation of $\mathcal{R}(F)$ around chance is simply due to noise then we would expect near chance (33%) classification accuracy. The leftmost point in Figure 2 shows the initial result, using boolean features in a log-linear classification model, where a test case involves guessing an L1 label for each individual sentence in the L2 corpus. While the accuracy does exceed chance, the margin is not very large.

One possible explanation for this small margin is that the language transfer signal is sparse, as it is likely that language transfer can only be used to correctly label a subset of L2 data. We test this by combining randomly sampled L2 sentences with the same L1 label, as shown along the horizontal axis of Figure 2. As the number of sentences used to create each test case is increased, we see an increase in accuracy that supports the argument for sparsity; if the features were simply weak predictors, this curve would be flat. The resulting margin is much larger, providing evidence that a significant portion of our features with $\mathcal{R}(F) < .66$ are not selected due to random noise in $\mathcal{R}$ and are indeed connected to language transfer.

The number and strength of these hypotheses is easily augmented with more data, as is the number of languages under consideration. Our results also motivate future work towards automatic generation of L1 targeted language education exercises, and the fact that TSG fragments are a component of a well studied generative language model makes them well suited to such generation tasks.

References

Daniel Blanchard, Joel Tetreault, Derrick Higgins, Aoife Cahill, and Martin Chodorow. 2013. Toefl11: A corpus of non-native english. Technical report, Educational Testing Service.

Julian Brooke and Graeme Hirst. 2012. Measuring Interlanguage: Native Language Identification with L1-influence Metrics. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*, pages 779–784, Istanbul, Turkey, May. European Language Resources Association (ELRA). ACL Anthology Identifier: L12-1016.

Glenn Carroll and Eugene Charniak. 1992. Two experiments on learning probabilistic dependency grammars from corpora. Technical Report CS-92-16, Brown University, Providence, RI, USA.

Trevor Cohn and Phil Blunsom. 2010. Blocked inference in bayesian tree substitution grammars. pages 225–230. Association for Computational Linguistics.

Aria Haghighi, Percy Liang, Taylor Berg-Kirkpatrick, and Dan Klein. 2008. Learning bilingual lexicons from monolingual corpora. In *ACL*, pages 771–779.

Scott Jarvis and Scott Crossley, editors. 2012. *Approaching Language Transfer Through Text Classification: Explorations in the Detection-based Approach*, volume 64. Multilingual Matters Limited, Bristol, UK.

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

Ryan T. McDonald, Joakim Nivre, Yvonne Quirmbach-Brundage, Yoav Goldberg, Dipanjan Das, Kuzman Ganchev, Keith Hall, Slav Petrov, Hao Zhang, Oscar Täckström, Claudia Bedini, Núria Bertomeu Castelló, and Jungmee Lee. 2013. Universal dependency annotation for multilingual parsing. In *ACL (2)*, pages 92–97.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Comput. Linguist.*, 29(1):19–51, March.

Matt Post and Daniel Gildea. 2009. Bayesian learning of a tree substitution grammar. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 45–48. Association for Computational Linguistics.

Ben Swanson and Eugene Charniak. 2013. Extracting the native language signal for second language acquisition. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 85–94, Atlanta, Georgia, June. Association for Computational Linguistics.

Joel Tetreault, Daniel Blanchard, and Aoife Cahill. 2013. A report on the first native language identification shared task. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, Atlanta, GA, USA, June. Association for Computational Linguistics.

Simple and Effective Approach for Consistent Training of Hierarchical Phrase-based Translation Models

Stephan Peitz[1] and **David Vilar**[2] and **Hermann Ney**[1]
[1] Lehrstuhl für Informatik 6
Computer Science Department
RWTH Aachen University
D-52056 Aachen, Germany
{peitz,ney}@cs.rwth-aachen.de

[2] Pixformance GmbH
D-10587 Berlin, Germany
david.vilar@gmail.com

Abstract

In this paper, we present a simple approach for consistent training of hierarchical phrase-based translation models. In order to consistently train a translation model, we perform hierarchical phrase-based decoding on training data to find derivations between the source and target sentences. This is done by synchronous parsing the given sentence pairs. After extracting k-best derivations, we reestimate the translation model probabilities based on collected rule counts. We show the effectiveness of our procedure on the IWSLT German→English and English→French translation tasks. Our results show improvements of up to 1.6 points BLEU.

1 Introduction

In state of the art statistical machine translation systems, the translation model is estimated by following heuristic: Given bilingual training data, a word alignment is trained with tools such as GIZA++ (Och and Ney, 2003) or *fast_align* (Dyer et al., 2013). Then, all valid translation pairs are extracted and the translation probabilities are computed as relative frequencies (Koehn et al., 2003).

However, this extraction method causes several problems. First, this approach does not consider, whether a translation pair is extracted from a likely alignment or not. Further, during the extraction process, models employed in decoding are not considered.

For phrase-based translation, a successful approach addressing these issues is presented in (Wuebker et al., 2010). By applying a phrase-based decoder on the source sentences of the training data and constraining the translations to the corresponding target sentences, k-best segmentations are produced. Then, the phrases used for these segmentations are extracted and counted. Based on the counts, the translation model probabilities are recomputed. To avoid over-fitting, leave-one-out is applied.

However, for hierarchical phrase-based translation an equivalent approach is still missing.

In this paper, we present a simple and effective approach for consistent reestimation of the translation model probabilities in a hierarchical phrase-based translation setup. Using a heuristically extracted translation model as starting point, the training data are parsed bilingually. From the resulting hypergraphs, we extract k-best derivations and the rules applied in each derivation. This is done with a top-down k-best parsing algorithm. Finally, the translation model probabilities are recomputed based on the counts of the extracted rules. In our procedure, we employ leave-one-out to avoid over-fitting. Further, we consider all models which are used in translation to ensure a consistent training.

Experimental results are presented on the German→English and English→French IWSLT shared machine translation task (Cettolo et al., 2013). We are able to gain improvements of up to 1.6% BLEU absolute and 1.4% TER over a competitive baseline. On all tasks and test sets, the improvements are statistically significant with at least 99% confidence.

The paper is structured as follow. First, we revise the state of the art hierarchical phrase-based extraction and translation process. In Section 3, we propose our training procedure. Finally, experimental results are given in Section 4 and we conclude with Section 5.

2 Hierarchical Phrase-based Translation

In hierarchical phrase-based translation (Chiang, 2005), discontinuous phrases with "gaps" are allowed. The translation model is formalized as a synchronous context-free grammar (SCFG)

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 174–179,
Gothenburg, Sweden, April 26-30 2014.

and consists of bilingual rules, which are based on bilingual standard phrases and discontinuous phrases. Each bilingual rule rewrites a generic non-terminal X into a pair of strings $\tilde{f}$ and $\tilde{e}$ with both terminals and non-terminals in both languages

$$X \rightarrow \langle \tilde{f}, \tilde{e} \rangle. \quad (1)$$

In a standard hierarchical phrase-based translation setup, obtaining these rules is based on a heuristic extraction from automatically word-aligned bilingual training data. Just like in the phrase-based approach, all bilingual rules of a sentence pair are extracted given an alignment. The standard phrases are stored as *lexical rules* in the rule set. In addition, whenever a phrase contains a sub-phrase, this sub-phrase is replaced by a generic non-terminal X. With these hierarchical phrases we can define the *hierarchical rules* in the SCFG. The rule probabilities which are in general defined as relative frequencies are computed based on the joint counts $C(X \rightarrow \langle \tilde{f}, \tilde{e} \rangle)$ of a bilingual rule $X \rightarrow \langle \tilde{f}, \tilde{e} \rangle$

$$p_H(\tilde{f}|\tilde{e}) = \frac{C(X \rightarrow \langle \tilde{f}, \tilde{e} \rangle)}{\sum_{\tilde{f}'} C(X \rightarrow \langle \tilde{f}', \tilde{e} \rangle)}. \quad (2)$$

The translation probabilities are computed in source-to-target as well as in target-to-source direction. In the translation processes, these probabilities are integrated in the log-linear combination among other models such as a language model, word lexicon models, word and phrase penalty and binary features marking hierarchical phrases, glue rule and rules with non-terminals at the boundaries.

The translation process of hierarchical phrase-based approach can be considered as parsing problem. Given an input sentence in the source language, this sentence is parsed using the source language part of the SCFG. In this work, we perform this step with a modified version of the CYK+ algorithm (Chappelier and Rajman, 1998). The output of this algorithm is a *hypergraph*, which represents all possible *derivations* of the input sentence. A derivation represents an application of rules from the grammar to generate the given input sentence. Using the the associated target part of the applied rule, for each derivation a translation can be constructed. In a second step, the language model score is incorporated. Given the hypergraph, this is done with the cube pruning algorithm presented in (Chiang, 2007).

3 Translation Model Training

We propose following pipeline for consistent hierarchical phrase-based training: First we train a word alignment, from which the baseline translation model is extracted as described in the previous section. The log-linear parameter weights are tuned with MERT (Och, 2003) on a development set to produce the baseline system. Next, we perform decoding on the training data. As the translations are constrained to the given target sentences, we name this step *forced decoding* in the following. Details are given in the next subsection. Given the counts $C_{FD}(X \rightarrow \langle \tilde{f}, \tilde{e} \rangle)$ of the rules, which have been applied in the forced decoding step, the translation probabilities $p_{FD}(\tilde{f}|\tilde{e})$ for the translation model are recomputed:

$$p_{FD}(\tilde{f}|\tilde{e}) = \frac{C_{FD}(X \rightarrow \langle \tilde{f}, \tilde{e} \rangle)}{\sum_{\tilde{f}'} C_{FD}(X \rightarrow \langle \tilde{f}', \tilde{e} \rangle)}. \quad (3)$$

Finally, using the translation model with the reestimated probabilities, we retune the log-linear parameter weights and obtain our final system.

3.1 Forced Decoding

In this section, we describe the forced decoding for hierarchical phrase-based translation in detail.

Given a sentence pair of the training data, we constrain the translation of the source sentence to produce the corresponding target sentence. For this constrained decoding process, the language model score is constant as the translation is fixed. Hence, the incorporation of the a language model is not needed. This results in a simplification of the decoding process as we do not have to employ the cube pruning algorithm as described in the previous section. Consequently, forced decoding for hierarchical phrase-based translation is equivalent to synchronous parsing of the training data. Dyer (2010) has described an approach to reduce the average-case run-time of synchronous parsing by splitting one bilingual parse into two successive monolingual parses. We adopt this method and first parse the source sentence and then the target sentence with CYK+.

If the given sentence pair has been parsed successfully, we employ a top-down k-best parsing algorithm (Chiang and Huang, 2005) on the resulting hypergraph to find the k-best derivations between the given source and target sentence. In this step, all models of the translation process are

included (except for the language model). Further, leave-one-out is applied to counteract overfitting. Note, that the model weights of the baseline system are used to perform forced decoding.

Finally, we extract and count the rules which have been applied in the derivations. These counts are used to recompute the translation probabilities.

3.2 Recombination

In standard hierarchical phrase-based decoding, partial derivations that are indistinguishable from each other are recombined. In (Huck et al., 2013) two schemes are presented. Either derivations that produce identical translations or derivations with identical language model context are recombined. As in forced decoding the translation is fixed and a language model is missing, both schemes are not suitable.

However, a recombination scheme is necessary to avoid derivations with the same application of rules. Further, recombining such derivations increases simultaneously the amounts of considered derivations during k-best parsing. Given two derivations with the same set of applied rules, the order of application of the rules may be different. Thus, we propose following scheme for recombining derivations in forced decoding: Derivations that produce identical sets of applied rules are recombined. Figure 1 shows an example for $k = 3$. Employing the proposed scheme, derivations d_1 and d_2 are recombined since both share the same set of applied rules ($\{r_1, r_3, r_2\}$).

$d_1:$	$\{r_1, r_3, r_2\}$	$d_1:$	$\{r_1, r_3, r_2\}$
$d_2:$	$\{r_3, r_2, r_1\}$	$d_3:$	$\{r_4, r_5, r_1, r_2\}$
$d_3:$	$\{r_4, r_5, r_1, r_2\}$	$d_4:$	$\{r_6, r_5, r_2, r_3\}$
	(a)		(b)

Figure 1: Example search space before (a) and after (b) applying recombination.

4 Experiments

4.1 Setup

The experiments were carried out on the IWSLT 2013 German→English shared translation task.[1]

[1] http://www.iwslt2013.org

	German	English	English	French
Sentences	4.32M		5.23M	
Run. Words	108M	109M	133M	147M
Vocabulary	836K	792K	845K	888K

Table 1: Statistics for the bilingual training data of the IWSLT 2013 German→English and English→French task.

It is focusing the translation of TED talks. Bilingual data statistics are given in Table 1. The baseline system was trained on all available bilingual data and used a 4-gram LM with modified Kneser-Ney smoothing (Kneser and Ney, 1995; Chen and Goodman, 1998), trained with the SRILM toolkit (Stolcke, 2002). As additional data sources for the LM we selected parts of the Shuffled News and LDC English Gigaword corpora based on cross-entropy difference (Moore and Lewis, 2010). In all experiments, the hierarchical search was performed as described in Section 2.

To confirm the efficacy of our approach, additional experiments were run on the IWSLT 2013 English→French task. Statistics are given in Table 1.

The training pipeline was set up as described in the previous section. Tuning of the log-linear parameter weights was done with MERT on a provided development set. As optimization criterion we used BLEU (Papineni et al., 2001).

Forced decoding was performed on the TED talks portion of the training data (∼140K sentences). In both tasks, around 5% of the sentences could not be parsed. In this work, we just skipped those sentences.

We report results in BLEU [%] and TER [%] (Snover et al., 2006). All reported results are averages over three independent MERT runs, and we evaluated statistical significance with *MultEval* (Clark et al., 2011).

4.2 Results

Figure 2 shows the performance of setups using translation models with reestimated translation probabilities. The setups vary in the k-best derivation size extracted in the forced decoding (fd) step. Based on the performance on the development set, we selected two setups with $k = 500$ using leave-one-out (+l1o) and $k = 750$ without leave-one-out (-l1o). Table 2 shows the final results for the German→English task. Performing consistent translation model training improves the translation

	dev*		eval11		test	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	33.1	46.8	35.7	44.1	30.5	49.7
forced decoding -l1o	33.2	**46.3**	**36.3**	**43.4**	**31.2**	**48.8**
forced decoding +l1o	**33.6**	**46.2**	**36.6**	**43.0**	**31.8**	**48.3**

Table 2: Results for the IWSLT 2013 German→English task. The development set used for MERT is marked with an asterisk (*). Statistically significant improvements with at least 99% confidence over the baseline are printed in boldface.

	dev*		eval11		test	
	BLEU[%]	TER[%]	BLEU[%]	TER[%]	BLEU[%]	TER[%]
baseline	28.1	55.7	37.5	42.7	31.7	49.5
forced decoding +l1o	**28.8**	**55.0**	**39.1**	**41.6**	**32.4**	**49.0**

Table 3: Results for the IWSLT 2013 English→French task. The development set used for MERT is marked with an asterisk (*). Statistically significant improvements with at least 99% confidence over the baseline are printed in boldface.

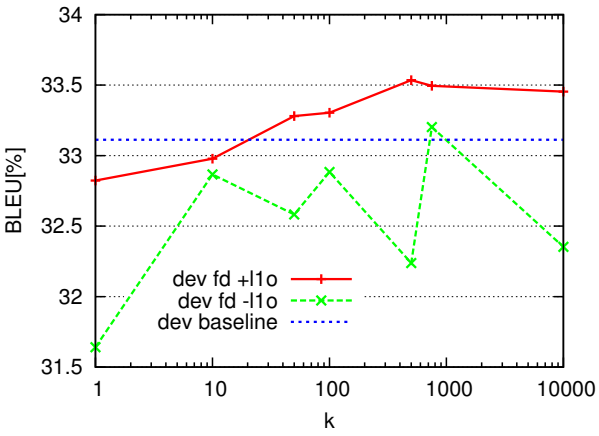


Figure 2: BLEU scores on the IWSLT German→English task of setups using translation models trained with different k-best derivation sizes. Results are reported on dev with (+l1o) and without leave-one-out (-l1o).

quality on all test sets significantly. We gain an improvement of up to 0.7 points in BLEU and 0.9 points in TER. Applying leave-one-out results in an additional improvement by up to 0.4 % BLEU and 0.5 % TER. The results for English→French are given in Table 3. We observe a similar improvement by up to 1.6 % BLEU and 1.1 % TER.

The improvements could be the effect of domain adaptation since we performed forced decoding on the TED talks portion of the training data. Thus, rules which were applied to decode the in-domain data might get higher translation probabilities.

Furthermore, employing leave-one-out seems to avoid overfitting as the average source rule length in training is reduced from 5.0 to 3.5 ($k = 500$).

5 Conclusion

We have presented a simple and effective approach for consistent training of hierarchical phrase-based translation models. By reducing hierarchical decoding on parallel training data to synchronous parsing, we were able to reestimate the translation probabilities including all models applied during the translation process. On the IWSLT German→English and English→French tasks, the final results show statistically significant improvements of up to 1.6 points in BLEU and 1.4 points in TER.

Our implementation was released as part of Jane (Vilar et al., 2010; Vilar et al., 2012; Huck et al., 2012; Freitag et al., 2014), the RWTH Aachen University open source statistical machine translation toolkit.[2]

Acknowledgments

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreements no 287658 and no 287755.

[2] http://www.hltpr.rwth-aachen.de/jane/

References

Mauro Cettolo, Jan Nieheus, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2013. Report on the 10th iwslt evaluation campaign. In *Proc. of the International Workshop on Spoken Language Translation*, Heidelberg, Germany, December.

J.-C. Chappelier and M. Rajman. 1998. A generalized CYK algorithm for parsing stochastic CFG. In *Proceedings of the First Workshop on Tabulation in Parsing and Deduction*, pages 133–137, April.

Stanley F. Chen and Joshuo Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Computer Science Group, Harvard University, Cambridge, MA, August.

David Chiang and Liang Huang. 2005. Better k-best Parsing. In *Proceedings of the 9th Internation Workshop on Parsing Technologies*, pages 53–64, October.

David Chiang. 2005. A Hierarchical Phrase-Based Model for Statistical Machine Translation. In *Proc. of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 263–270, Ann Arbor, Michigan, June.

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

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *49th Annual Meeting of the Association for Computational Linguistics:shortpapers*, pages 176–181, Portland, Oregon, June.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A Simple, Fast, and Effective Reparameterization of IBM Model 2. In *Proceedings of NAACL-HLT*, pages 644–648, Atlanta, Georgia, June.

Chris Dyer. 2010. Two monolingual parses are better than one (synchronous parse). In *In Proc. of HLT-NAACL*.

Markus Freitag, Matthias Huck, and Hermann Ney. 2014. Jane: Open source machine translation system combination. In *Conference of the European Chapter of the Association for Computational Linguistics*, Gothenburg, Sweden, April. To appear.

Matthias Huck, Jan-Thorsten Peter, Markus Freitag, Stephan Peitz, and Hermann Ney. 2012. Hierarchical Phrase-Based Translation with Jane 2. *The Prague Bulletin of Mathematical Linguistics*, 98:37–50, October.

Matthias Huck, David Vilar, Markus Freitag, and Hermann Ney. 2013. A performance study of cube pruning for large-scale hierarchical machine translation. In *Proceedings of the NAACL 7th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 29–38, Atlanta, Georgia, USA, June.

Reinerd Kneser and Hermann Ney. 1995. Improved backing-off for M-gram language modeling. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processingw*, volume 1, pages 181–184, May.

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

R.C. Moore and W. Lewis. 2010. Intelligent Selection of Language Model Training Data. In *ACL (Short Papers)*, pages 220–224, Uppsala, Sweden, July.

Franz Josef Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51, March.

Franz Josef Och. 2003. Minimum Error Rate Training in Statistical Machine Translation. In *Proc. of the 41th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 160–167, Sapporo, Japan, July.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2001. Bleu: a Method for Automatic Evaluation of Machine Translation. IBM Research Report RC22176 (W0109-022), IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598, September.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA, August.

Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. In *Proc. of the Int. Conf. on Speech and Language Processing (ICSLP)*, volume 2, pages 901–904, Denver, CO, September.

David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. 2010. Jane: Open source hierarchical translation, extended with reordering and lexicon models. In *ACL 2010 Joint Fifth Workshop on Statistical Machine Translation and Metrics MATR*, pages 262–270, Uppsala, Sweden, July.

David Vilar, Daniel Stein, Matthias Huck, and Hermann Ney. 2012. Jane: an advanced freely available hierarchical machine translation toolkit. *Machine Translation*, 26(3):197–216, September.

Joern Wuebker, Arne Mauser, and Hermann Ney. 2010. Training phrase translation models with leaving-one-out. In *Proceedings of the 48th Annual Meeting of the Assoc. for Computational Linguistics*, pages 475–484, Uppsala, Sweden, July.

Some Experiments with a Convex IBM Model 2

Andrei Simion
Columbia University
IEOR Department
New York, NY, 10027
aas2148@columbia.edu

Michael Collins
Columbia University
Computer Science
New York, NY, 10027
mc3354@columbia.edu

Clifford Stein
Columbia University
IEOR Department
New York, NY, 10027
cs2035@columbia.edu

Abstract

Using a recent convex formulation of IBM Model 2, we propose a new initialization scheme which has some favorable comparisons to the standard method of initializing IBM Model 2 with IBM Model 1. Additionally, we derive the Viterbi alignment for the convex relaxation of IBM Model 2 and show that it leads to better F-Measure scores than those of IBM Model 2.

1 Introduction

The IBM translation models are widely used in modern statistical translation systems. Unfortunately, apart from Model 1, the IBM models lead to non-convex objective functions, leading to methods (such as EM) which are not guaranteed to reach the global maximum of the log-likelihood function. In a recent paper, Simion et al. introduced a convex relaxation of IBM Model 2, I2CR-2, and showed that it has performance on par with the standard IBM Model 2 (Simion et al., 2013).

In this paper we make the following contributions:

- We explore some applications of I2CR-2. In particular, we show how this model can be used to seed IBM Model 2 and compare the speed/performance gains of our initialization under various settings. We show that initializing IBM Model 2 with a version of I2CR-2 that uses large batch size yields a method that has similar run time to IBM Model 1 initialization and at times has better performance.

- We derive the Viterbi alignment for I2CR-2 and compare it directly with that of IBM Model 2. Previously, Simion et al. (2013) had compared IBM Model 2 and I2CR-2 by using IBM Model 2's Viterbi alignment rule, which is not necessarily the optimal alignment for I2CR-2. We show that by comparing I2CR-2 with IBM Model 2 by using each model's optimal Viterbi alignment the convex model consistently has a higher F-Measure. F-Measure is an important metric because it has been shown to be correlated with BLEU scores (Marcu et al., 2006).

Notation. We adopt the notation introduced in (Och and Ney, 2003) of having 1^m2^n denote the training scheme of m IBM Model 1 EM iterations followed by initializing Model 2 with these parameters and running n IBM Model 2 EM iterations. The notation $\text{EG}_B^m2^n$ means that we run m iterations of I2CR-2's EG algorithm (Simion et al., 2013) with batch size of B, initialize IBM Model 2 with I2CR-2's parameters, and then run n iterations of Model 2's EM.

2 The IBM Model 1 and 2 Optimization Problems

In this section we give a brief review of IBM Models 1 and 2 and the convex relaxation of Model 2, I2CR-2 (Simion et al., 2013). The standard approach in training parameters for Models 1 and 2 is EM, whereas for I2CR-2 an exponentiated-gradient (EG) algorithm was developed (Simion et al., 2013).

We assume that our set of training examples is $(e^{(k)}, f^{(k)})$ for $k = 1 \ldots n$, where $e^{(k)}$ is the k'th English sentence and $f^{(k)}$ is the k'th French sentence. The k'th English sentence is a sequence of words $e_1^{(k)} \ldots e_{l_k}^{(k)}$ where l_k is the length of the k'th English sentence, and each $e_i^{(k)} \in E$; similarly the k'th French sentence is a sequence $f_1^{(k)} \ldots f_{m_k}^{(k)}$ where each $f_j^{(k)} \in F$. We define $e_0^{(k)}$ for $k = 1 \ldots n$ to be a special NULL word (note that E contains the NULL word). IBM Model 2 is detailed in several sources such as (Simion et al., 2013) and (Koehn, 2004).

The convex and non-convex objectives of respectively IBM Model 1 and 2 can be found in (Simion

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 180–184,
Gothenburg, Sweden, April 26-30 2014.

et al., 2013). For I2CR-2, the convex relaxation of IBM Model 2, the objective is given by

$$\frac{1}{2n}\sum_{k=1}^{n}\sum_{j=1}^{m_k}\log'\sum_{i=0}^{l_k}\frac{t(f_j^{(k)}|e_i^{(k)})}{(L+1)} + \frac{1}{2n}\sum_{k=1}^{n}\sum_{j=1}^{m_k}\log'\sum_{i=0}^{l_k}\min\{t(f_j^{(k)}|e_i^{(k)}), d(i|j)\}\ .$$

For smoothness reasons, Simion et al. (2013) defined $\log'(z) = \log(z+\lambda)$ where $\lambda = .001$ is a small positive constant. The I2CR-2 objective is a convex combination of the convex IBM Model 1 objective and a direct (convex) relaxation of the IBM2 Model 2 objective, and hence is itself convex.

3 The Viterbi Alignment for I2CR-2

Alignment models have been compared using methods other than Viterbi comparisons; for example, Simion et al. (2013) use IBM Model 2's optimal rule given by (see below) Eq. 2 to compare models while Liang et al. (2006) use posterior decoding. Here, we derive and use I2CR-2's Viterbi alignment. To get the Viterbi alignment of a pair $(e^{(k)}, f^{(k)})$ using I2CR-2 we need to find $a^{(k)} = (a_1^{(k)}, \ldots, a_{m_k}^{(k)})$ which yields the highest probability $p(f^{(k)}, a^{(k)}|e^{(k)})$. Referring to the I2CR-2 objective, this corresponds to finding $a^{(k)}$ that maximizes

$$\frac{\log\prod_{j=1}^{m_k} t(f_j^{(k)}|e^{(k)}_{a_j^{(k)}})}{2} + \frac{\log\prod_{j=1}^{m_k}\min\{t(f_j^{(k)}|e^{(k)}_{a_j^{(k)}}), d(a_j^{(k)}|j)\}}{2}\ .$$

Putting the above terms together and using the monotonicity of the logarithm, the above reduces to finding the vector $a^{(k)}$ which maximizes

$$\prod_{j=1}^{m_k} t(f_j^{(k)}|e^{(k)}_{a_j^{(k)}})\min\{t(f_j^{(k)}|e^{(k)}_{a_j^{(k)}}), d(a_j^{(k)}|j)\}.$$

As with IBM Models 1 and 2, we can find the vector $a^{(k)}$ by splitting the maximization over the components of $a^{(k)}$ and focusing on finding $a_j^{(k)}$ given by

$$\text{argmax}_a(t(f_j^{(k)}|e_a^{(k)})\min\{t(f_j^{(k)}|e_a^{(k)}), d(a|j)\})\ . \quad (1)$$

In previous experiments, Simion et al. (Simion et al., 2013) were comparing I2CR-2 and IBM Model 2 using the standard alignment formula derived in a similar fashion from IBM Model 2:

$$a_j^{(k)} = \text{argmax}_a(t(f_j^{(k)}|e_a^{(k)})d(a|j))\ . \quad (2)$$

4 Experiments

In this section we describe experiments using the I2CR-2 optimization problem combined with the stochastic EG algorithm (Simion et al., 2013) for parameter estimation. The experiments conducted here use a similar setup to those in (Simion et al., 2013). We first describe the data we use, and then describe the experiments we ran.

4.1 Data Sets

We use data from the bilingual word alignment workshop held at HLT-NAACL 2003 (Michalcea and Pederson, 2003). We use the Canadian Hansards bilingual corpus, with 247,878 English-French sentence pairs as training data, 37 sentences of development data, and 447 sentences of test data (note that we use a randomly chosen subset of the original training set of 1.1 million sentences, similar to the setting used in (Moore, 2004)). The development and test data have been manually aligned at the word level, annotating alignments between source and target words in the corpus as either "sure" (S) or "possible" (P) alignments, as described in (Och and Ney, 2003).

As a second data set, we used the Romanian-English data from the HLT-NAACL 2003 workshop consisting of a training set of 48,706 Romanian-English sentence-pairs, a development set of 17 sentence pairs, and a test set of 248 sentence pairs.

We carried out our analysis on this data set as well, but because of space we only report the details on the Hansards data set. The results on the Romanian data were similar, but the magnitude of improvement was smaller.

4.2 Methodology

Our experiments make use of either standard training or intersection training (Och and Ney, 2003). For standard training, we run a model in the source-target direction and then derive the alignments on the test or development data. For each of the

Training	2^{10}	$1^5 2^{10}$	$\text{EG}^1_{125} 2^{10}$	$\text{EG}^1_{1250} 2^{10}$
Iteration		Objective		
0	-224.0919	-144.2978	-91.2418	-101.2250
1	-110.6285	-85.6757	-83.3255	-85.5847
2	-91.7091	-82.5312	-81.3845	-82.1499
3	-84.8166	-81.3380	-80.6120	-80.9610
4	-82.0957	-80.7305	-80.2319	-80.4041
5	-80.9103	-80.3798	-80.0173	-80-1009
6	-80.3620	-80.1585	-79.8830	-79.9196
7	-80.0858	-80.0080	-79.7911	-79.8048
8	-79.9294	-79.9015	-79.7247	-79.7284
9	-79.8319	-79.8240	-79.6764	-79.6751
10	-79.7670	-79.7659	-79.6403	-79.6354

Table 1: Objective results for the English → French IBM Model 2 seeded with either uniform parameters, IBM Model 1 ran for 5 EM iterations, or I2CR-2 ran for 1 iteration with either B = 125 or 1250. Iteration 0 denotes the starting IBM 2 objective depending on the initialization.

models—IBM Model 1, IBM Model 2, and I2CR-2— we apply the conventional methodology to intersect alignments: first, we estimate the t and d parameters using models in both source-target and target-source directions; second, we find the most likely alignment for each development or test data sentence in each direction; third, we take the intersection of the two alignments as the final output from the model. For the I2CR-2 EG (Simion et al., 2013) training, we use batch sizes of either $B = 125$ or $B = 1250$ and a step size of $\gamma = 0.5$ throughout.

We measure the performance of the models in terms of *Precision*, *Recall*, *F-Measure*, and *AER* using only sure alignments in the definitions of the first three metrics and sure and possible alignments in the definition of AER, as in (Simion et al., 2013) and (Marcu et al., 2006). For our experiments, we report results in both AER (lower is better) and F-Measure (higher is better).

4.3 Initialization and Timing Experiments

We first report the summary statistics on the test set using a model trained only in the English-French direction. In these experiments we seeded IBM Model 2's parameters either with those of IBM Model 1 run for 5, 10 or 15 EM iterations or I2CR-2 run for 1 iteration of EG with a batch size of either $B = 125$ or 1250. For uniform comparison, all of our implementations were written in C++ using STL/Boost containers.

There are several takeaways from our experiments, which are presented in Table 2. We first note that with $B = 1250$ we get higher F-Measure and lower AER even though we use less training time: 5 iterations of IBM Model 1 EM training takes about 3.3 minutes, which is about the time it takes for 1 iteration of EG with a batch size of 125 (4.1 minutes); on the other hand, using $B = 1250$ takes EG 1.7 minutes and produces the best results across almost all iterations. Additionally, we note that the initial solution given to IBM Model 2 by running I2CR-2 for 1 iteration with $B = 1250$ is fairly strong and allows for further progress: IBM2 EM training improves upon this solution during the first few iterations. We also note that this behavior is global: no IBM 1 initialization scheme produced subsequent solutions for IBM 2 with as low in AER or high in F-Measure. Finally, comparing Table 1 which lists objective values with Table 2 which lists alignment statistics, we see that although the objective progression is similar throughout, the alignment quality is different.

To complement the above, we also ran intersection experiments. Seeding IBM Model 2 by Model 1 and intersecting the alignments produced by the English-French and French-English models gave both AER and F-Measure which were better than those that we obtained by any seeding of IBM Model 2 with I2CR-2. However, there are still reasons why I2CR-2 would be useful in this context. In particular, we note that I2CR-2 takes roughly half the time to progress to a better solution than IBM Model 1 run for 5 EM iterations. Second, a possible remedy to the above loss in marginal improvement when taking intersections would be to use a more refined method for obtaining the joint alignment of the English-French and French-English models, such as "grow-diagonal" (Och and Ney, 2003).

4.4 Viterbi Comparisons

For the decoding experiments, we used IBM Model 1 as a seed to Model 2. To train IBM Model 1, we follow (Moore, 2004) and (Och and Ney, 2003) in running EM for 5, 10 or 15 iterations. For the EG algorithm, we initialize all parameters uniformly and use 10 iterations of EG with a batch size of 125. Given the lack of development data for the alignment data sets, for both IBM Model 2 and the I2CR-2 method, we report test set F-Measure and AER results for each of the 10 iterations, rather than picking the results from a single iteration.

Training	2^{10}	$1^5 2^{10}$	$1^{10} 2^{10}$	$1^{15} 2^{10}$	$\text{EG}^1_{125} 2^{10}$	$\text{EG}^1_{1250} 2^{10}$
Iteration			AER			
0	0.8713	0.3175	0.3177	0.3160	**0.2329**	0.2662
1	0.4491	0.2547	0.2507	0.2475	0.2351	**0.2259**
2	0.2938	0.2428	0.2399	0.2378	0.2321	**0.2180**
3	0.2593	0.2351	0.2338	0.2341	0.2309	**0.2176**
4	0.2464	0.2298	0.2305	0.2310	0.2283	**0.2168**
5	0.2383	0.2293	0.2299	0.2290	0.2268	**0.2188**
6	0.2350	0.2273	0.2285	0.2289	0.2274	**0.2205**
7	0.2320	0.2271	0.2265	0.2286	0.2274	**0.2213**
8	0.2393	0.2261	0.2251	0.2276	0.2278	**0.2223**
9	0.2293	0.2253	0.2246	0.2258	0.2284	**0.2217**
10	0.2288	0.2248	0.2249	0.2246	0.2275	**0.2223**
Iteration			F-Measure			
0	0.0427	0.5500	0.5468	0.5471	**0.6072**	0.5977
1	0.4088	0.5846	0.5876	0.5914	0.6005	**0.6220**
2	0.5480	0.5892	0.5916	0.5938	0.5981	**0.6215**
3	0.5750	0.5920	0.5938	0.5947	0.5960	**0.6165**
4	0.5814	0.5934	0.5839	0.5952	0.5955	**0.6129**
5	0.5860	0.5930	0.5933	0.5947	0.5945	**0.6080**
6	0.5873	0.5939	0.5936	0.5940	0.5924	**0.6051**
7	0.5884	0.5931	0.5955	0.5941	0.5913	**0.6024**
8	0.5899	0.5932	0.5961	0.5942	0.5906	**0.6000**
9	0.5899	0.5933	0.5961	0.5958	0.5906	**0.5996**
10	0.5897	0.5936	0.5954	0.5966	0.5910	**0.5986**

Table 2: Results on the Hansards data for English → French IBM Model 2 seeded using different methods. The first three columns are for a model seeded with IBM Model 1 ran for 5, 10 or 15 EM iterations. The fourth and fifth columns show results when we seed with I2CR-2 ran for 1 iteration either with $B = 125$ or 1250. Iteration 0 denotes the starting statistics.

Training	$1^5 2^{10}$	$1^{10} 2^{10}$	$1^{15} 2^{10}$	EG^{10}_{125}	EG^{10}_{125}
Viterbi Rule	$t \times d$	$t \times d$	$t \times d$	$t \times d$	$t \times \min\{t \times d\}$
Iteration			AER		
0	**0.2141**	0.2159	0.2146	0.9273	0.9273
1	0.1609	0.1566	**0.1513**	0.1530	0.1551
2	0.1531	0.1507	0.1493	0.1479	**0.1463**
3	0.1477	0.1471	0.1470	0.1473	**0.1465**
4	0.1458	**0.1444**	0.1449	0.1510	0.1482
5	0.1455	0.1438	**0.1435**	0.1501	0.1482
6	0.1436	0.1444	**0.1429**	0.1495	0.1481
7	0.1436	**0.1426**	0.1435	0.1494	0.1468
8	0.1449	**0.1427**	0.1437	0.1508	0.1489
9	0.1454	**0.1426**	0.1430	0.1509	0.1481
10	0.1451	0.1430	**0.1423**	0.1530	0.1484
Iteration			F-Measure		
0	**0.7043**	0.7012	0.7021	0.0482	0.0482
1	0.7424	0.7477	0.7534	0.7395	**0.7507**
2	0.7468	0.7499	0.7514	0.7448	**0.7583**
3	0.7489	0.7514	0.7520	0.7455	**0.7585**
4	0.7501	0.7520	0.7516	0.7418	**0.7560**
5	0.7495	0.7513	0.7522	0.7444	**0.7567**
6	0.7501	0.7501	0.7517	0.7452	**0.7574**
7	0.7493	0.7517	0.7507	0.7452	**0.7580**
8	0.7480	0.7520	0.7504	0.7452	**0.7563**
9	0.7473	0.7511	0.7513	0.7450	**0.7590**
10	0.7474	0.7505	0.7520	0.7430	**0.7568**

Table 3: Intersected results on the English-French data for IBM Model 2 and I2CR-2 using either IBM Model 1 trained to 5, 10, or 15 EM iterations to seed IBM2 and using either the IBM2 or I2CR-2 Viterbi formula for I2CR-2.

In Table 3 we report F-Measure and AER results for each of the iterations under IBM Model 2 and I2CR-2 models using either the Model 2 Viterbi rule of Eq. 2 or I2CR-2's Viterbi rule in Eq. 1. We note that unlike in the previous experiments presented in (Simion et al., 2013), we are directly testing the quality of the alignments produced by I2CR-2 and IBM Model 2 since we are getting the Viterbi alignment for each model (for completeness, we also have included in the fourth column the Viterbi alignments we get by using the IBM Model 2 Viterbi formula with the I2CR-2 parameters as Simion et al. (2013) had done previously). For these experiments we report intersection statistics. Under its proper decoding formula, I2CR-2 model yields a higher F-Measure than any setting of IBM Model 2. Since AER and BLEU correlation is arguably known to be weak while F-Measure is at times strongly related with BLEU (Marcu et al., 2006), the above results favor the convex model.

We close this section by pointing out that the main difference between the IBM Model 2 Viterbi rule of Eq. 2 and the I2CR-2 Viterbi rule in Eq. 1 is that the Eq. 1 yield fewer alignments when doing intersection training. Even though there are fewer alignments produced, the quality in terms of F-Measure is better.

5 Conclusions and Future Work

In this paper we have explored some of the details of a convex formulation of IBM Model 2 and showed it may have an application either as a new initialization technique for IBM Model 2 or as a model in its own right, especially if the F-Measure is the target metric. Other possible topics of interest include performing efficient sensitivity analysis on the I2CR-2 model, analyzing the balance between the IBM Model 1 and I2CR-1 (Simion et al., 2013) components of the I2CR-2 objective, studying I2CR-2's intersection training performance using methods such as "grow diagonal" or "agreement" (Liang et al., 2006), and integrating it into the GIZA++ open source library so we can see how much it affects the downstream system.

Acknowledgments

Michael Collins and Andrei Simion are partly supported by NSF grant IIS-1161814. Cliff Stein is

partly supported by NSF grants CCF-0915681 and CCF-1349602. We thank Professor Paul Blaer and Systems Engineer Radu Sadeanu for their help setting up some of the hardware used for these experiments. We also thank the anonymous reviewers for many useful comments; we hope to pursue the comments we were not able to address in a followup paper.

References

Peter L. Bartlett, Ben Taskar, Michael Collins and David Mcallester. 2004. Exponentiated Gradient Algorithms for Large-Margin Structured Classification. *In Proceedings of NIPS*.

Steven Boyd and Lieven Vandenberghe. 2004. Convex Optimization. Cambridge University Press.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert. L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics*, 19:263-311.

Michael Collins, Amir Globerson, Terry Koo, Xavier Carreras and Peter L. Bartlett. 2008. Exponentiated Gradient Algorithms for Conditional Random Fields and Max-Margin Markov Networks. *Journal Machine Learning,* 9(Aug): 1775-1822.

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum Likelihood From Incomplete Data via the EM Algorithm. *Journal of the royal statistical society, series B*, 39(1):1-38.

Alexander Fraser and Daniel Marcu. 2007. Measuring Word Alignment Quality for Statistical Machine Translation. *Journal Computational Linguistics*, 33(3): 293-303.

Joao V. Graca, Kuzman Ganchev and Ben Taskar. 2007. Expectation Maximization and Posterior Constraints. *In Proceedings of NIPS.*

Yuhong Guo and Dale Schuurmans. 2007. Convex Relaxations of Latent Variable Training. *In Proceedings of NIPS.*

Simon Lacoste-Julien, Ben Taskar, Dan Klein, and Michael Jordan. 2008. Word Alignment via Quadratic Assignment. *In Proceedings of the HLT-NAACL.*

Phillip Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. *In Proceedings of the EMNLP*.

Phillip Koehn. 2008. Statistical Machine Translation. Cambridge University Press.

Kivinen, J., Warmuth, M. 1997. Exponentiated Gradient Versus Gradient Descent for Linear Predictors. *Information and Computation*, 132, 1-63.

Percy Liang, Ben Taskar and Dan Klein. 2006. Alignment by Agreement. *In Proceedings of NAACL.*

Daniel Marcu, Wei Wang, Abdessamad Echihabi, and Kevin Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. *In Proceedings of the EMNLP*.

Rada Michalcea and Ted Pederson. 2003. An Evaluation Exercise in Word Alignment. *HLT-NAACL 2003: Workshop in building and using Parallel Texts: Data Driven Machine Translation and Beyond.*

Robert C. Moore. 2004. Improving IBM Word-Alignment Model 1. *In Proceedings of the ACL.*

Stephan Vogel, Hermann Ney and Christoph Tillman. 1996. HMM-Based Word Alignment in Statistical Translation. *In Proceedings of COLING*.

Franz Och and Hermann Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational-Linguistics*, 29(1): 19-52.

Andrei Simion, Michael Collins and Cliff Stein. 2013. A Convex Alternative to IBM Model 2. *In Proceedings of the EMNLP*.

Kristina Toutanova and Michel Galley. 2011. Why Initialization Matters for IBM Model 1: Multiple Optima and Non-Strict Convexity. *In Proceedings of the ACL.*

Ashish Vaswani, Liang Huang and David Chiang. 2012. Smaller Alignment Models for Better Translations: Unsupervised Word Alignment with the $L0$-norm. *In Proceedings of the ACL.*

Active Learning for Post-Editing Based Incrementally Retrained MT

Aswarth Dara Josef van Genabith Qun Liu John Judge Antonio Toral
School of Computing
Dublin City University
Dublin, Ireland
{adara,josef,qliu,jjudge,atoral}@computing.dcu.ie

Abstract

Machine translation, in particular statistical machine translation (SMT), is making big inroads into the localisation and translation industry. In typical workflows (S)MT output is checked and (where required) manually post-edited by human translators. Recently, a significant amount of research has concentrated on capturing human post-editing outputs as early as possible to incrementally update/modify SMT models to avoid repeat mistakes. Typically in these approaches, MT and post-edits happen sequentially and chronologically, following the way unseen data (the translation job) is presented. In this paper, we add to the existing literature addressing the question whether and if so, to what extent, this process can be improved upon by Active Learning, where input is not presented chronologically but dynamically selected according to criteria that maximise performance with respect to (whatever is) the remaining data. We explore novel (source side-only) selection criteria and show performance increases of 0.67-2.65 points TER absolute on average on typical industry data sets compared to sequential PE-based incrementally retrained SMT.

1 Introduction and Related Research

Machine Translation (MT) has evolved dramatically over the last two decades, especially since the appearance of statistical approaches (Brown et al., 1993). In fact, MT is nowadays succesfully used in the localisation and translation industry, as for many relevant domains such as technical documentation, post-editing (PE) of MT output by human translators (compared to human translation from scratch) results in notable productivity gains, as a number of industry studies have shown convincingly, e.g. (Plitt and Masselot, 2010). Furthermore, incremental retraining and update techniques (Bertoldi et al., 2013; Levenberg et al., 2010; Mathur et al., 2013; Simard and Foster, 2013) allow these PEs to be fed back into the MT model, resulting in an MT system that is continuously updated to perform better on forthcoming sentences, which should lead to a further increase in productivity.

Typically, post-editors are presented with MT output units (sentences) in the order in which input sentences appear one after the other in the translation job. Because of this, incremental MT retraining and update models based on PE outputs also proceed in the same chronological order determined by the input data. This may be suboptimal. In this paper we study the application of Active Learning (AL) to the scenario of PE MT and subsequent PE-based incremental retraining. AL selects data (here translation inputs and their MT outputs for PE) according to criteria that maximise performance with respect to the remaining data and may diverge from processing data items in chronological order. This may allow incrementally PE-based retrained MT to (i) improve more rapidly than chronologically PE-based retrained MT and (ii) result in overall productivity gains.

The main contributions of this paper include:

- Previous work (Haffari et al., 2009; Bloodgood and Callison-Burch, 2010) shows that, given a (static) training set, AL can improve the quality of MT. By contrast, here we show that AL-based data selection for human PE improves incrementally and dynamically retrained MT, reducing overall PE time of translation jobs in the localisation industry application scenarios.
- We propose novel selection criteria for AL-based PE: we adapt cross-entropy difference (Moore and Lewis, 2010; Axelrod et al., 2011), originally used for domain adaptation, and propose an extension to cross entropy difference with a vocabulary saturation filter (Lewis and Eetemadi, 2013).
- While much of previous work concentrates on research datasets (e.g. Europarl, News Commentary), we use industry data (techni-

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 185–189,
Gothenburg, Sweden, April 26-30 2014.

cal manuals). (Bertoldi et al., 2013) shows that the repetition rate of news is considerably lower than that of technical documentation, which impacts on the results obtained with incremental retraining.

- Unlike in previous research, our AL-based selection criteria take into account only the source side of the data. This supports selection before translation, keeping costs to a minimum, a priority in commercial PE MT applications.
- Our experiments show that AL-based selection works for PE-based incrementally retrained MT with overall performance gains around 0.67 to 2.65 TER absolute on average.

AL has been successfully applied to many tasks in natural language processing, including parsing (Tang et al., 2002), named entity recognition (Miller et al., 2004), to mention just a few. See (Olsson, 2009) for a comprehensie overview of the application of AL to natural language processing. (Haffari et al., 2009; Bloodgood and Callison-Burch, 2010) apply AL to MT where the aim is to build an optimal MT model from a given, static dataset. To the best of our knowledge, the most relevant previous research is (González-Rubio et al., 2012), which applies AL to interactive MT. In addition to differences in the AL selection criteria and data sets, our goals are fundamentally different: while the previous work aimed at reducing human effort in interactive MT, we aim at reducing the overall PE time in PE-based incremental MT update applications in the localisation industry.

In our experiments reported in Section 3 below we want to explore a space consisting of a considerable number of selection strategies and incremental retraining batch sizes. In order to be able to do this, we use the target side of our industry translation memory data to approximate human PE output and automatic TER (Snover et al., 2006) scores as a proxy for human PE times (O'Brien, 2011).

2 Methodology

Given a translation job, our goal is to reduce the overall PE time. At each stage, we select sentences that are given to the post editor in such a way that uncertain sentences (with respect to the MT system at hand)[1] are post-edited first. We then translate the n top-ranked sentences using the MT system and use the human PEs of the MT outputs to retrain the system. Algorithm 1 describes our method, where s and t stand for source and target, respectively.

Algorithm 1 Sentence Selection Algorithm

```
Input:
L ⟵ Initial training data
M ⟵ Initial MT model
for C ∈ (Random,Sequential,Ngram,CED,CEDN) do
   U ⟵ Translation job
   while size(U) > 0 do
      U1.s ⟵ SelectTopSentences(C, U.s)
      U1[1].t ⟵ Translate(M, U1.s)
      U1.t ⟵ PostEdit(U1[1].t)
      U ⟵ U - U1
      L ⟵ L ∪ U1
      M ⟵ TrainModel (L)
   end while
end for
```

We use two baselines, i.e. random and sequential. In the random baseline, the batch of sentences at each iteration are selected randomly. In the sequential baseline, the batches of sentences follow the same order as the data.

Aside from the *Random* and *Sequential* baselines we use the following selection criteria:

- **N-gram Overlap.** An SMT system will encounter problems translating sentences containing n-grams not seen in the training data. Thus, PEs of sentences with high number of unseen n-grams are considered to be more informative for updating the current MT system. However, for the MT system to translate unseen n-grams accurately, they need to be seen a minimum number V times.[2] We use an n-gram overlap function similar to the one described in (González-Rubio et al., 2012) given in Equation 1 where $N(T^{(i)})$ and $N(S^{(i)})$ return i-grams in training data and the sentence S, respectively.

$$unseen(S) = \frac{\sum_{i=1}^{n} \{|N(T^{(i)}) \cap N(S^{(i)})| > V\}}{\sum_{i=1}^{n} N(S^{(i)})} \quad (1)$$

- **Cross Entropy Difference (CED).** This metric is originally used in data selection (Moore and Lewis, 2010; Axelrod et al., 2011). Given an in-domain corpus I and a general corpus O, language models are built from both,[3] and each sentence in O is scored according to the entropy H difference (Equation

[1] The uncertainty of a sentence with respect to the model can be measured according to different criteria, e.g. percentage of unknown n-grams, perplexity etc.

[2] Following (González-Rubio et al., 2012) we use $V = 10$.

[3] In order to make the LMs comparable they have the same size. As commonly the size of O is considerable bigger than I, this means that the LM for O is built from a subset of the same size as I.

2). The lower the score given to a sentence, the more useful it is to train a system for the specific domain I.

$$score(s) = H_I(s) - H_O(s) \quad (2)$$

In our AL scenario, we have the current training corpus *L* and an untranslated corpus *U*. CED is applied to select sentences from *U* that are (i) different from *L* (as we would like to add sentences that add new information to the model) and (ii) similar to the overall corpus *U* (as we would like to add sentences that are common in the untranslated data). Hence we apply CED and select sentences from *U* that have high entropy with respect to *L* and low entropy with respect to *U* (Equation 3).

$$score(s) = H_U(s) - H_L(s) \quad (3)$$

- **CED + n-gram (CEDN).** This is an extension of the CED criterion inspired by the concept of the vocabulary saturation filter (Lewis and Eetemadi, 2013). CED may select many very similar sentences, and thus it may be the case that some of them are redundant. By post-processing the selection made by CED with vocabulary saturation we aim to spot and remove redudant sentences. This works in two steps. In the first step, all the sentences from *U* are scored using the CED metric. In the second step, we down-rank sentences that are considered redundant. The top sentence is selected, and its n-grams are stored in *localngrams*. For the remaining sentences, one by one, their n-grams are matched against *localngrams*. If the intersection between them is lower than a predefined threshold, the current sentence is added and *localngrams* is updated with the n-grams from the current sentence. Otherwise the sentence is down-ranked to the bottom. In our experiments, the value n = 1 produces best results.

3 Experiments and Results

We use technical documentation data taken from Symantec translation memories for the English–French (EN–FR) and English–German (EN–DE) language pairs (both directions) for our experiments. The statistics of the data (training and incremental splits) are shown in Table 1.

All the systems are trained using the Moses (Koehn et al., 2007) phrase-based statistical MT system, with IRSTLM (Federico et al., 2008) for language modelling (n-grams up to order five) and with the alignment heuristic *grow-diag-final-and*.

For the experiments, we considered two settings for each language pair in each direction. In the first setting, the initial MT system is trained using the training set (39,679 and 54,907 sentence pairs for EN–FR and EN–DE, respectively). Then, a batch of 500 source sentences is selected from the incremental dataset according to each of the selection criteria, and translations are obtained with the initial MT system. These translations are post-edited and the corrected translations are added to the training data.[4] We then train a new MT system using the updated training data (initial training data plus PEs of the first batch of sentences). The updated model will be used to translate the next batch. The same process is repeated until the incremental dataset is finished (16 and 20 iterations for English–French and English–German, respectively). For each batch we compute the TER score between the MT output and the refererence translations for the sentences of that batch. We then compute the average TER score for all the batches. These average scores, for each selection criterion, are reported in Table 2.

In the second setting, instead of using the whole training data, we used a subset of (randomly selected) 5,000 sentence pairs for training the initial MT system and a subset of 20,000 sentences from the remaining data as the incremental dataset. Here we take batches of 1,000 sentences (thus 20 batches). The results are shown in Table 3.

The first setting aims to reflect the situation where a translation job is to be completed for a domain for which we have a considerable amount of data available. Conversely, the second setting reflects the situation where a translation job is to be carried out for a domain with little (if any) available data.

Dir	Random	Seq.	Ngram	CED	CEDN
EN→FR	29.64	29.81	**28.97**	29.25	29.05
FR→EN	27.08	27.04	**26.15**	26.63	26.39
EN→DE	24.00	24.08	22.34	22.60	**22.32**
DE→EN	19.36	19.34	17.70	17.97	**17.48**

Table 2: TER average scores for Setting 1

Dir	Random	Seq.	Ngram	CED	CEDN
EN→FR	36.23	36.26	35.20	35.48	**35.17**
FR→EN	33.26	33.34	32.26	32.69	**32.17**
EN→DE	32.23	32.19	30.58	31.96	**29.98**
DE→EN	27.24	27.29	26.10	26.73	**24.94**

Table 3: TER average scores for Setting 2

For Setting 1 (Table 2), the best result is obtained by the CEDN criterion for two out of the four directions. For EN→FR, n-gram overlap

[4] As this study simulates the post-editing, we use the references of the translated segments instead of the PEs.

Type	EN–FR			EN–DE		
	Sentences	*Avg. EN SL*	*Avg. FR SL*	*Sentences*	*Avg. EN SL*	*Avg. DE SL*
Training	39,679	13.55	15.28	54,907	12.66	12.90
Incremental	8,000	13.74	15.50	10,000	12.38	12.61

Table 1: Data Statistics for English–French and English–German Symantec Translation Memory Data. SL stands for sentence length, EN stands for English, FR stands for French and DE stands for German

performs slightly better than CEDN (0.08 points lower) with a decrease of 0.67 and 0.84 points when compared to the baselines (random and sequential, respectively). For FR→EN, n-gram overlap results in a decrease of 0.93 and 0.89 points compared to the baselines. The decrease in average TER score is higher for the EN→DE and for DE→EN directions, i.e. 1.68 and 1.88 points respectively for CEDN compared to the random baseline.

In the scenario with limited data available beforehand (Table 3), CEDN is the best performing criterion for all the language directions. For the EN–FR and FR–EN language pairs, CEDN results in a decrease of 1.06 and 1.09 points compared to the random baseline. Again, the decrease is higher for the EN–DE and DE–EN language pairs, i.e. 2.25 and 2.30 absolute points on average.

Figure 1 shows the TER scores per iteration for each of the criteria, for the scenario DE→EN Setting 2 (the trends are similar for the other scenarios). The two baselines exhibit slight improvement over the iterations, both starting at around .35 TER points and finishing at around .25 points. Conversely, all the three criteria start at very high scores (in the range [.5,.6]) and then improve considerably to arrive at scores below .1 for the last iterations. Compared to Ngram and CED, CEDN reaches better scores earlier on, being the criterion with the lowest score up to iteration 13.

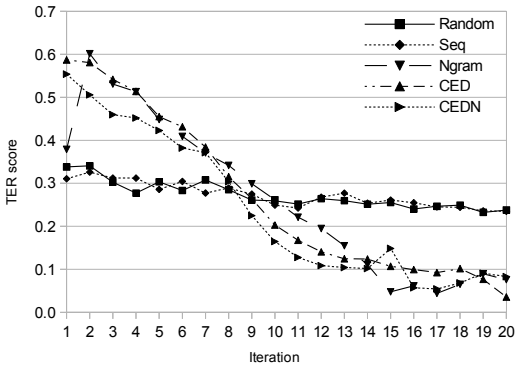


Figure 1: Results per iteration, DE→EN Setting 2

Figure 1 together with Tables 2 and 3 show that AL for PE-based incremental MT retraining really works: all AL based methods (Ngram, CED, CEDN) show strong improvements over both baselines after the initial 8-9 iterations (Figure 1) and best performance on the complete incremental data sets, resulting in a noticeable decrease of the overall TER score (Tables 2 and 3). In six out of eight scenarios, our novel metric CEDN obtains the best result.

4 Conclusions and Future Work

This paper has presented an application of AL to MT for dynamically selecting automatic translations of sentences for human PE, with the aim of reducing overall PE time in a PE-based incremental MT retraining scenario in a typical industrial localisation workflow that aims to capitalise on human PE as early as possible to avoid repeat mistakes.

Our approach makes use of source side information only, uses two novel selection criteria based on cross entropy difference and is tested on industrial data for two language pairs. Our best performing criteria allow the incrementally retrained MT systems to improve their performance earlier and reduce the overall TER score by around one and two absolute points for English–French and English–German, respectively.

In order to be able to explore a space of selection criteria and batch sizes, our experiments simulate PE, in the sense that we use the target reference (instead of PEs) and approximate PE time with TER. Given that TER correlates well with PE time (O'Brien, 2011), we expect AL-based selection of sentences for human PE to lead to overall reduction of PE time. In the future work, we plan to do the experiments using PEs to retrain the system and measuring PE time.

In this work, we have taken batches of sentences (size 500 to 1,000) and do full retraining. As future work, we plan to use fully incremental retraining and perform the selection on a sentence-by-sentence basis (instead of taking batches).

Finally and importantly, a potential drawback of our approach is that by dynamically selecting individual sentences for PE, the human post-editor looses context, which they may use if processing sentences sequentially. We will explore the trade off between the context lost and the productivity gain achieved, and ways of supplying context (e.g. previous and following sentence) for real PE.

Acknowledgements

This work is supported by Science Foundation Ireland (Grants 12/TIDA/I2438, 07/CE/I1142 and 12/CE/I2267) as part of the Centre for Next Generation Localisation (www.cngl.ie) at Dublin City University. We would like to thank Symantec for the provision of data sets used in our experiments.

References

Amittai Axelrod, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 355–362, Stroudsburg, PA, USA. Association for Computational Linguistics.

Nicola Bertoldi, Mauro Cettolo, and Marcello Federico. 2013. Cache-based online adaptation for machine translation enhanced computer assisted translation. In *Proceedings of the XIV Machine Translation Summit*, pages 35–42, Nice, France.

Michael Bloodgood and Chris Callison-Burch. 2010. Bucking the trend: Large-scale cost-focused active learning for statistical machine translation. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *ACL*, pages 854–864. The Association for Computer Linguistics.

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

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. IRSTLM: an open source toolkit for handling large scale language models. In *INTERSPEECH*, pages 1618–1621. ISCA.

Jesús González-Rubio, Daniel Ortiz-Martínez, and Francisco Casacuberta. 2012. Active learning for interactive machine translation. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 245–254, Stroudsburg, PA, USA. Association for Computational Linguistics.

Gholamreza Haffari, Maxim Roy, and Anoop Sarkar. 2009. Active learning for statistical phrase-based machine translation. In *HLT-NAACL*, pages 415–423. The Association for Computational Linguistics.

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

Abby Levenberg, Chris Callison-Burch, and Miles Osborne. 2010. Stream-based translation models for statistical machine translation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 394–402, Stroudsburg, PA, USA. Association for Computational Linguistics.

William Lewis and Sauleh Eetemadi. 2013. Dramatically reducing training data size through vocabulary saturation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 281–291, Sofia, Bulgaria, August. Association for Computational Linguistics.

Prashant Mathur, Mauro Cettolo, and Marcello Federico. 2013. Online learning approaches in computer assisted translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation, ACL*, pages 301–308, Sofia, Bulgaria.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *Proceedings of HLT*, pages 337–342.

Robert C. Moore and William Lewis. 2010. Intelligent selection of language model training data. In *Proceedings of the ACL 2010 Conference Short Papers*, ACLShort '10, pages 220–224, Stroudsburg, PA, USA. Association for Computational Linguistics.

Sharon O'Brien. 2011. Towards predicting post-editing productivity. *Machine Translation*, 25(3):197–215, September.

Fredrik Olsson. 2009. A literature survey of active machine learning in the context of natural language processing. Technical Report T2009:06.

Mirko Plitt and François Masselot. 2010. A productivity test of statistical machine translation post-editing in a typical localisation context. *Prague Bull. Math. Linguistics*, 93:7–16.

Michel Simard and George Foster. 2013. Pepr: Post-edit propagation using phrase-based statistical machine translation. In *Proceedings of the XIV Machine Translation Summit*, pages 191–198, Nice, France.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*, pages 223–231, Cambridge, MA.

Min Tang, Xiaoqiang Luo, and Salim Roukos. 2002. Active learning for statistical natural language parsing. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 120–127, Stroudsburg, PA, USA. Association for Computational Linguistics.

Analysis and Prediction of Unalignable Words in Parallel Text

Frances Yung **Kevin Duh** **Yuji Matsumoto**
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192 Japan
pikyufrances-y|kevinduh|matsu@is.naist.jp

Abstract

Professional human translators usually do not employ the concept of word alignments, producing translations *'sense-for-sense'* instead of *'word-for-word'*. This suggests that unalignable words may be prevalent in the parallel text used for machine translation (MT). We analyze this phenomenon in-depth for Chinese-English translation. We further propose a simple and effective method to improve automatic word alignment by pre-removing unalignable words, and show improvements on hierarchical MT systems in both translation directions.

1 Motivation

It is generally acknowledged that absolute equivalence between two languages is impossible, since concept lexicalization varies across languages. Major translation theories thus argue that texts should be translated *'sense-for-sense'* instead of *'word-for-word'* (Nida, 1964). This suggests that unalignable words may be an issue for the parallel text used to train current statistical machine translation (SMT) systems. Although existing automatic word alignment methods have some mechanism to handle the lack of exact word-for-word alignment (e.g. null probabilities, fertility in the IBM models (Brown et al., 1993)), they may be too coarse-grained to model the 'sense-for-sense' translations created by professional human translators.

For example, the Chinese term *'tai-yang'* literally means *'sun'*, yet the concept it represents is equivalent to the English term *'the sun'*. Since the concept of a definite article is not incorporated in the morphology of *'tai yang'*, the added *'the'* is not aligned to any Chinese word. Yet in another context like *'the man'*, *'the'* can be the translation of the Chinese demonstrative pronoun *'na'*, literally means *'that'*. A potential misunderstanding is that unalignable words are simply function words; but from the above example, we see that whether a word is alignable depends very much on the concept and the linguistic context.

As the quantity and quality of professionally-created parallel text increase, we believe there is a need to examine the question of unalignable words in-depth. Our goal is to gain a better understanding of what makes a fluent human translation and use this insight to build better word aligners and MT systems. Our contributions are two-fold:

1) We analyze 13000 sentences of manually word-aligned Chinese-English parallel text, quantifying the characteristics of unalignable words.

2) We propose a simple and effective way to improve automatic word alignment, based on predicting unalignable words and temporarily removing them during the alignment training procedure.

2 Analysis of Unalignable Words

Our manually-aligned data, which we call ORACLE data, is a Chinese-to-English corpus released by the LDC (Li et al., 2010)[1]. It consists of ∼13000 Chinese sentences from *news* and *blog* domains and their English translation . English words are manually aligned with the Chinese characters. Characters without an exact counterpart are annotated with categories that state the functions of the words. These characters are either aligned to 'NULL', or attached to their dependency heads, if any, and aligned together to form a multi-word alignment. For example, *'the'* is annotated as [DET], for 'determiner', and aligned to *'tai-yang'* together with *'sun'*.

In this work, any English word or Chinese character without an exact counterpart are called *unalignable words*, since they are not core to the

[1] LDC2012T16, LDC2012T20 and LDC2012T24

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 190–194,
Gothenburg, Sweden, April 26-30 2014.

	word types	unalignable tokens	core tokens
core or unalignable	3581 (12%)	146,693 (17%)	562,801 (66%)
always core	25320 (88%)	/	147,373 (17%)

Table 1: Number of core and unalignable words in hand aligned ORACLE corpus

Most frequent *unalignable* word types	Token count	
	Chinese	English
Top 50	34,987 (68%)	83,905 (88%)
Top 100	40,121 (78%)	89,609 (94%)

Table 2: Count of *unalignable* words by types

multi-word alignment. All other English words or Chinese characters are referred to as *core words*.

2.1 What kind of words are unalignable?

Analyzing the hand aligned corpus, we find that words annotated as unalignable do not come from a distinct list. Table 1 reveals that 88% of the word types are unambiguously core words. Yet these word types, including singletons, account for only 17% of the word tokens. On the other hand, another 17% of the total word tokens are annotated as unalignable. So, most word types are possibly unalignable but only in a small portion of their occurrence, such as the following examples:

(1a) Chi: *yi* [*ge*] *di fang*
one (measure word) place
Eng: *one* *place*

(1b) Chi: *ge ren*
personal
Eng: *personal*

(2a) Chi: *ming tian* *zhong wu*
(tomorrow) (midday)
Eng: *tomorrow* [*at*] *midday*

(2b) Chi: *zai* *jia*
at/in/on home
Eng: *at* *home*

In example (1a), *'ge'* is a measure word that is exclusive in Chinese, but in (1b), it is part of the multiword unit *'ge-ren'* for *'personal'*. Similarly, prepositions, such as *'at'*, can either be omitted or translated depending on context.

Nonetheless, unalignable words are by no means evenly distributed among word types. Table 2 shows that the top 100 most frequent unalignable word types already covers 78% and 94% of all Chinese and English unalignable instances, respectively. Word type is thus an important clue.

Intuitively, words with POS defined only in one of the languages are likely to be unalignable. To examine this, we automatically tagged the ORACLE data using the Standford Tagger (Toutanova et al., 2003). We find that the unalignable words include all POS categories of either language, though indeed some POS are more frequent. Table 3 lists the top 5 POS categories that most unalignable words belong to and the percentage they are annotated as unalignable. Some POS categories like DEG are mostly unalignable regardless of context, but other POS tags such as DT and IN depend on context.

Chi. POS	**No. and % of unalign.**	**Eng. POS**	**No. and % of unalign.**
DEG	7411(97%)	DT	27715 (75%)
NN	6138 (4%)	IN	19303 (47%)
AD	6068 (17%)	PRP	5780 (56%)
DEC	5572 (97%)	TO	5407 (62%)
VV	4950 (6%)	CC	4145 (36%)

Table 3: Top 5 POS categories of Chinese and English unalignable words

Note also that many Chinese unalignable words are nouns (NN) and verbs (VV). Clearly we cannot indiscriminately consider all nouns as unalignable. Some examples of unalignable content words in Chinese are:

(3) Chi: *can jia* *hui jian* [*huo dong*]
participate meeting activity
Eng: *participate* *in the meeting*

(4) Chi: *hui yi* *de* *yuan man* [*ju xing*]
meeting 's successful take place
Eng: *success* *of* *the meeting*

English verbs and adjectives are often nominalized to abstract nouns (such as *'meeting'* from *'meet'*, or *'success'* from *'succeed'*), but such derivation is rare in Chinese morphology. Since POS is not morphologically marked in Chinese, *'meeting'* and *'meet'* are the same word. To reduce the processing ambiguity and produce more natural translation, extra content words are added to mark the nominalization of abstract concepts. For example, *'hui jian'* is originally *'to meet'*. Adding *'huo dong'*(activity) transforms it to a noun phrase

(example 3), similar to the the addition of *'ju sing'*(take place) to the adjective *'yuan man'* (example 4). These unalignable words are not lexically dependent but are inferred from the context, and thus do not align to any source words.

To summarize, a small number of word types cover 17% of word tokens that are unalignable, but whether these words are unalignable depends significantly on context. Although there is no list of *'always unalignable'* words types or POS categories, our analysis shows there are regularities that may be exploited by an automatic classifier.

3 Improved Automatic Word Alignment

We first propose a classifier for predicting whether a word is unalignable. Let (e_1^J, f_1^K) be a pair of sentence with length J and K. For each word in (e_1^J, f_1^K) that belongs to a predefined list[2] of potentially unalignable words, we run a binary classifier. A separate classifier is built for each word type in the list, and an additional classifier for all the remaining words in each language.

We train an SVM classifier based on the following features: **Local context:** Unigrams and POS in window sizes of 1, 3, 5, 7 around the word in question. **Top token-POS pairs:** This feature is defined by whether the token in question and its POS tag is within the top *n* frequent token-POS pairs annotated as unalignable like in Tables 2 and 3. Four features are defined with $n = 10, 30, 50, 100$. Since the top frequent unalignable words cover most of the counts as shown in the previous analysis, being in the top *n* list is a strong positive features. **Number of likely unalignable words per sentence:** We hypothesize that the translator will not add too many tokens to the translation and delete too many from the source sentence. In the ORACLE data, 68% sentences have more than 2 unalignable words. We approximate the number of likely unalignable words in the sentence by counting the number of words within the top 100 token-POS pairs annotated as unalignable. **Sentence length and ratio:** Longer sentences are more likely to contain unalignable words than shorter sentences. Also sentence ratios that deviate significantly from the mean are likely to contain unalignable words. **Presence of alignment candidate:** This is a negative feature defined by whether there is an alignment candidate in the target sentence for the source word in question, or vice versa. The candidates are extracted from the top *n* frequent words aligned to a particular word according to the manual alignments of the ORACLE data. Five features are defined with $n = 5, 10, 20, 50, 100$ and one 'without limit', such that a more possible candidate will be detected by more features.

Next, we propose a simple yet effective modification to the word alignment training pipeline:

1. Predict unalignable words by the classifier
2. Remove these words from the training corpus
3. Train word alignment model (e.g. GIZA++)[3]
4. Combine the word alignments in both directions with heuristics (grow-diag-final-and)
5. Restore unaligned words to original position
6. Continue with rule extraction and the rest of the MT pipeline.

The idea is to reduce the difficulty for the word alignment model by removing unaligned words.

4 End-to-End Translation Experiments

In our experiments, we first show that removing manually-annotated unaligned words in ORACLE data leads to improvements in MT of both translation directions. Next, we show how a classifier trained on ORACLE data can be used to improve MT in another large-scale un-annotated dataset.[4]

4.1 Experiments on ORACLE data

We first performed an ORACLE experiment using gold standard unaligned word labels. Following the training pipeline in Section 3, we removed gold unalignable words before running GIZA++ and restore them afterwards. 90% of the data is used for alignment and MT training, while 10% of the data is reserved for testing.

The upper half of Table 4 list the alignment precision, recall and F1 of the resulting alignments, and quality of the final MT outputs. **Baseline** is the standard MT training pipeline without removal of unaligned words. Our **Proposed** approach performs better in alignment, phrase-based (PBMT) and hierarchical (Hiero) systems. The results, evaluated by BLEU, METEOR and TER, support our hypothesis that removing gold unalignable words helps improve word alignment and the resulting SMT.

[2]We define the list as the top 100 word types with the highest count of unalignable words per language according to the hand annotated data.

[3]We can suppress the NULL probabilities of the model.

[4]All experiments are done using standard settings for Moses PBMT and Hiero with 4-gram LM and mslr-bidirectional-fe reordering (Koehn et al., 2007). The classifier is trained using LIBSVM (Chang and Lin, 2011).

	Align acc.		PBMT C-E	PBMT E-C	Hiero C-E	Hiero E-C
ORACLE	P .711	B	11.4	17.4	10.3	15.8
Baseline	R .488	T	**70.9**	69.0	75.9	72.3
	F1.579	M	21.8	23.9	21.08	23.7
ORACLE	P **.802**	B	**11.8**$^+$	**18.3**$^+$	**11.0**$^+$	**17.2**$^+$
Proposed	R **.509**	T	71.4$^-$	**65.7**$^+$	**74.7**$^+$	**68.7**$^+$
(gold)	F1.**623**	M	**22.1**$^+$	**24.1**$^+$	**22.0**$^+$	**24.0**$^+$
REAL		B	18.2	**18.5**	17.0	17.2
Baseline		T	**63.4**	67.2	68.0	71.4
		M	22.9	**24.6**	22.9	**24.8**
REAL		B	**18.6**	18.5	**17.6**$^+$	**18.1**$^+$
Proposed		T	63.8$^-$	**66.5**$^+$	**67.6**	**69.7**$^+$
(predict)		M	**23.2**$^+$	24.5	**23.4**$^+$	24.7

Table 4: MT results of ORACLE and REAL experiments. Highest score per metric is bolded. $\{^+/^-\}$ indicates statistically significant improvement/degradation, $p < 0.05$. (P: precision; R: recall; B: BLEU; M: METEOR; T:TER)

For comparison, a naive classifier that labels all top-30 token-POS combinations as unalignable performs poorly as expected (PBMT BLEU: 9.87 in C-E direction). We also evaluated our proposed classifier on this task: the accuracy is 92% and it achieves BLEU of 11.55 for PBMT and 10.84 for Hiero in C-E direction, which is between the results of gold-unalign and baseline.

4.2 Experiments on large-scale REAL data

We next performed a more realistic experiment: the classifier trained on ORACLE data is used to automatically label a large data, which is then used to train a MT system. This REAL data consists of parallel text from the NIST OpenMT2008.[5] MT experiments are performed in both directions.

The lower half of Table 4 shows the performance of the resulting MT systems. We observe that our proposed approach is still able to improve over the baseline. In particular, Hiero achieved statistical significant improvements in BLEU and METEOR. [6] Comparing to the results of PBMT, this suggests our method may be most effective in improving systems where rule extraction is sensitive to the underlying alignments, such as Hiero and Syntax-based MT. Table 5 shows the lexical translations for some rare Chinese words: the baseline tends to incorrectly align these to function words (garbage collection), while the proposed method's translations are more reasonable.

[5]We use the standard MT08 test sets; the training data includes LDC2004T08, 2005E47, 2005T06, 2007T23, 2008T06, 2008T08, 2008T18, 2009T02, 2009T06, 2009T15, and 2010T03 (34M English words and 1.1M sentences). Since we do not have access to all OpenMT data, e.g. FBIS, our results may not be directly comparable to other systems in the evaluation.

[6]Interestingly, PBMT did better than Hiero in this setup.

Chinese word	English lexical translation	
	Baseline only	Propose only
xie (bring)	him	bringing
xing (form)	and	model
dan (but)	it, the, they	yet, nevertheless
pa (scare)	that, are, be	fears, worried

Table 5: Examples of translations exclusively found in the top **15** lexical translation.

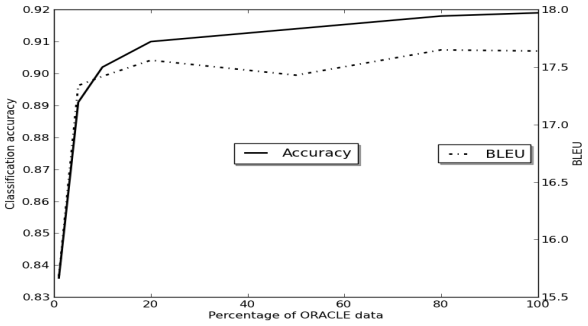


Figure 1: Classifier accuracy and MT results V.S. proportion of ORACLE data

To evaluate how much annotation is needed for the classifier, we repeat experiments using different proportions of the ORACLE data. Figure 1 shows training by 20% of the data (2600 sents.) already leads to significant improvements ($p < 0.05$), which is a reasonable annotation effort.

5 Conclusion

We analyzed in-depth the phenomenon of unalignable words in parallel text, and show that what is unalignable depends on the word's concept and context. We argue that this is not a trivial problem, but with an unalignable word classifier and a simple modified MT training pipeline, we can achieve small but significant gains in end-to-end translation. In related work, the issue of dropped pronouns (Chung and Gildea, 2010) and function words (Setiawan et al., 2010; Nakazawa and Kurohashi, 2012) have been found important in word alignment, and (Fossum et al., 2008) showed that syntax features are helpful for fixing alignments. An interesting avenue of future work is to integrate these ideas with ours, in particular by exploiting syntax and viewing unalignable words as aligned at a structure above the lexical level.

References

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

Chih-Chung Chang and Chih-Jen Lin. 2011. Libsvm : a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(27).

Tagyoung Chung and Daniel Gildea. 2010. Effects of empty categories on machine translation. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*.

Victoria Fossum, Kevin Knight, and Steven Abney. 2008. Using syntax to improve word alignment precision for syntax-based machine translation. *Proceedings of the Workshop on Statistical Machine Translation*.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Xuansong Li, Niyu Ge, Stephen Grimes, Stephanie M. Strassel, and Kazuaki Maeda. 2010. Enriching word alignment with linguistic tags. *Proceedings of International Conference on Language Resources and Evaluation*.

Toshiaki Nakazawa and Sado Kurohashi. 2012. Alignment by bilingual generation and monolingual derivation. *Proceedings of the International Conference on Computational Linguistics*.

Eugene A Nida. 1964. *Toward a Science of Translating: with Special Reference to Principles and Procedures Involved in Bible Translating*. BRILL.

Hendra Setiawan, Chris Dyer, and Philip Resnik. 2010. Discriminative word alignment with a function word reordering model. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*.

Kristina Toutanova, Dan Klein, Christopher Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Enhancing Authorship Attribution By Utilizing Syntax Tree Profiles

Michael Tschuggnall and Günther Specht
Institute of Computer Science, University of Innsbruck
Technikerstraße 21a, 6020 Innsbruck, Austria
{michael.tschuggnall, guenther.specht}@uibk.ac.at

Abstract

The aim of modern authorship attribution approaches is to analyze known authors and to assign authorships to previously unseen and unlabeled text documents based on various features. In this paper we present a novel feature to enhance current attribution methods by analyzing the grammar of authors. To extract the feature, a syntax tree of each sentence of a document is calculated, which is then split up into length-independent patterns using pq-grams. The mostly used pq-grams are then used to compose sample profiles of authors that are compared with the profile of the unlabeled document by utilizing various distance metrics and similarity scores. An evaluation using three different and independent data sets reveals promising results and indicate that the grammar of authors is a significant feature to enhance modern authorship attribution methods.

1 Introduction

The increasing amount of documents available from sources like publicly available literary databases often raises the question of verifying disputed authorships or assigning authors to unlabeled text fragments. The original problem was initiated already in the midst of the twentieth century by Mosteller and Wallace, who tried to find the correct authorships of *The Federalist Papers* (Mosteller and Wallace, 1964), nonetheless authorship attribution is still a major research topic. Especially with latest events in politics and academia, the verification of authorships becomes increasingly important and is used frequently in areas like juridical applications (*Forensic Linguistics*) or cybercrime detection (Nirkhi and Dharaskar, 2013). Similarily to works in the field of plagiarism detection (e.g. (Stamatatos, 2009; Tschuggnall and Specht, 2013b)) which aim to find text fragments not written but claimed to be written by an author, the problem of traditional authorship attribution is defined as follows: Given several authors with text samples for each of them, the question is to label an unknown document with the correct author. In contrast to this so-called *closed-class* problem, an even harder task is addressed in the *open-class* problem, where additionally a "none-of-them"-answer is allowed (Juola, 2006).

In this paper we present a novel feature for the traditional, closed-class authorship attribution task, following the assumption that different authors have different writing styles in terms of the grammar structure that is used mostly unconsciously. Due to the fact that an author has many different choices of how to formulate a sentence using the existing grammar rules of a natural language, the assumption is that the way of constructing sentences is significantly different for individual authors. For example, the famous Shakespeare quote *"To be, or not to be: that is the question."* (S1) could also be formulated as *"The question is whether to be or not to be."* (S2) or even *"The question is whether to be or not."* (S3) which is semantically equivalent but differs significantly according to the syntax (see Figure 1). The main idea of this approach is to quantify those differences by calculating grammar profiles for each candidate author as well as for the unlabeled document, and to assign one of the candidates as the author of the unseen document by comparing the profiles. To quantify the differences between profiles multiple metrics have been implemented and evaluated.

The rest of this paper is organized as follows: Section 2 sketches the main idea of the algorithm which incorporates the distance metrics explained in detail in Section 3. An extensive evaluation us-

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 195–199,
Gothenburg, Sweden, April 26-30 2014.

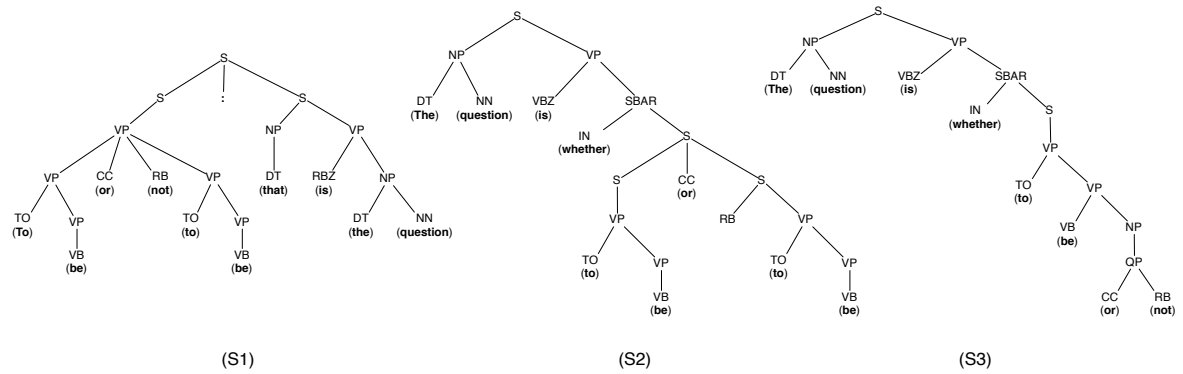


Figure 1: Syntax Trees Resulting From Parsing Sentence (S1), (S2) and (S3).

ing three different test sets is shown in Section 4, while finally Section 5 and Section 6 summarize related work and discuss future work, respectively.

2 Syntax Tree Profiles

The basic idea of the approach is to utilize the syntax that is used by authors to distinguish authorships of text documents. Based on our previous work in the field of intrinsic plagiarism detection (Tschuggnall and Specht, 2013c; Tschuggnall and Specht, 2013a) we modify and enhance the algorithms and apply them to be used in closed-class authorship attribution.

The number of choices an author has to formulate a sentence in terms of grammar is rather high, and the assumption in this approach is that the concrete choice is made mostly intuitively and unconsciously. Evaluations shown in Section 4 reinforce that solely parse tree structures represent a significant feature that can be used to distinguish between authors.

From a global view the approach comprises the following three steps: *(A)* Creating a grammar profile for each author, *(B)* creating a grammar profile for the unlabeled document, and *(C)* calculating the distance between each author profile and the document profile and assigning the author having the lowest distance (or the highest similarity, depending on the distance metric chosen). As this approach is based on profiles a key criterion is the creation of distinguishable author profiles. In order to calculate a grammar profile for an author or a document, the following procedure is applied: *(1)* Concatenate all text samples for the author into a single, large sample document, *(2)* split the resulting document into single sentences and calculate a syntax tree for each sentence, *(3)* calculate the pq-gram index for each tree, and *(4)* compose the final grammar profile from the normalized frequencies of pq-grams.

At first the concatenated document is cleaned to contain alphanumeric characters and punctuation marks only, and then split into single sentences[1]. Each sentence is then parsed[2]. For example, Figure 1 depicts the syntax trees resulting from sentences (S1), (S2) and (S3). The labels of each tree correspond to a Penn Treebank tag (Marcus et al., 1993), where e.g *NP* corresponds to a noun phrase or *JJS* to a superlative adjective. In order to examine solely the structure of sentences, the terminal nodes (words) are ignored.

Having computed a syntax tree for every sentence, the pq-gram index (Augsten et al., 2010) of each tree is calculated in the next step. Pq-grams consist of a stem (p) and a base (q) and may be related to as "n-grams for trees". Thereby p defines how much nodes are included vertically, and q defines the number of nodes to be considered horizontally. For example, a pq-gram using $p = 2$ and $q = 3$ starting from level two of tree (S1) would be `[S-VP-VP-CC-RB]`. In order to obtain all pq-grams of a tree, the base is additionally shifted left and right: If then less than p nodes exist horizontally, the corresponding place in the pq-gram is filled with $\star$, indicating a missing node. Applying this idea to the previous example, also the pq-grams `[S-VP-*-*-VP]` (base shifted left by two), `[S-VP-*-VP-CC]` (base shifted left by one), `[S-VP-RB-VP-*]` (base shifted right by one) and `[S-VP-VP-*-*]` (base shifted right by two) have to be considered. Finally, the pq-gram index contains all pq-grams of

[1] using OpenNLP, http://incubator.apache.org/opennlp, visited October 2013

[2] using the Stanford Parser (Klein and Manning, 2003)

a syntax tree, whereby multiple occurences of the same pq-grams are also present multiple times in the index.

The remaining part for creating the author profile is to compute the pq-gram index of the whole document by combining all pq-gram indexes of all sentences. In this step the number of occurences is counted for each pq-gram and then normalized by the total number of all appearing pq-grams. As an example, the three mostly used pq-grams of a selected document together with their normalized frequencies are {[NP-NN-⋆-⋆-⋆], 2.7%}, {[PP-IN-⋆-⋆-⋆], 2.3%}, and {[S-VP-⋆-⋆-VBD], 1.1%}. The final *pq-gram profile* then consists of the complete table of pq-grams and their occurences in the given document.

3 Distance and Similarity Metrics

With the use of the syntax tree profiles calculated for each candidate author as well as for the unlabeled document, the last part is to calculate a distance or similarity, respectively, for every author profile. Finally, the unseen document is simply labeled with the author of the best matching profile.

To investigate on the best distance or similarity metric to be used for this approach, several metrics for this problem have been adapted and evaluated[3]: 1. CNG (Kešelj et al., 2003), 2. Stamatatos-CNG (Stamatatos, 2009), 3. Stamatatos-CNG with Corpus Norm (Stamatatos, 2007), 4. Sentence-SPI.

For the latter, we modified the original SPI score (Frantzeskou et al., 2006) so that each sentence is traversed separately: Let S_D be the set of sentences of the document, $I(s)$ the pq-gram-index of sentence s and P_x the profile of author X, then the Sentence-SPI score is calculated as follows:

$$s_{P_x,P_D} = \sum_{s \in S_D} \sum_{p \in I(s)} \begin{cases} 1 & \text{if } p \in P_x \\ 0 & \text{else} \end{cases}$$

4 Evaluation

The approach described in this paper has been extensively evaluated using three different English data sets, whereby all sets are completely unrelated and of different types: (1.) CC04: the training set used for the Ad-hoc-Authorship Attribution Competition workshop held in 2004[4] - type: novels, authors: 4, documents: 8, samples per author: 1; (2.) FED: the (undisputed) federalist papers written by Hamilton, Madison and Jay in the 18th century - type: political essays, authors: 3, documents: 61, samples per author: 3; (3.) PAN12: from the state-of-the-art corpus, especially created for the use in authorship identification for the PAN 2012 workshop[5] (Juola, 2012), all closed-classed problems have been chosen - type: misc, authors: 3-16, documents: 6-16, samples per author: 2.
For the evaluation, each of the sets has been used to optimize parameters while the remaining sets have been used for testing. Besides examining the discussed metrics and values for p and q (e.g. by choosing $p = 1$ and $q = 0$ the pq-grams of a grammar profile are equal to pure POS tags), two additional optimization variables have been integrated for the similarity metric Sentence-SPI:

- **topPQGramCount t_c:** by assigning a value to this parameter, only the corresponding amount of mostly used pq-grams of a grammar profile are used.

- **topPQGramOffset t_o:** based on the idea that all authors might have a frequently used and common set of syntax rules that are predefined by a specific language, this parameter allows to ignore the given amount of mostly used pq-grams. For example if $t_o = 3$ in Table 1, the first pq-gram to be used would be [NP-NNP-⋆-⋆-⋆].

The evaluation results are depicted in Table 1. It shows the rate of correct author attributions based on the grammar feature presented in this paper.

Generally, the algorithm worked best using the *Sentence-SPI* score, which led to a rate of 72% by using the PAN12 data set for optimization. The optimal configuration uses $p = 3$ and $q = 2$, which is the same configuration that was used in (Augsten et al., 2010) to produce the best results. The highest scores are gained by using a limit of top pq-grams ($t_c \sim 65$) and by ignoring the first three pq-grams ($t_o = 3$), which indicates that it is sufficient to limit the number of syntax structures

[3]The algorithm names are only used as a reference for this paper, but were not originally proposed like that

[4]http://www.mathcs.duq.edu/∼juola/authorship_contest.html, visited Oct. 2013

[5]PAN is a well-known workshop on Uncovering Plagiarism, Authorship, and Social Software Misuses. http://pan.webis.de, visited Oct. 2013

metric	p	q	Optimized With	CC04	FED	PAN12	Overall
Sentence-SPI ($t_c = 65, t_o = 3$)	3	2	PAN12	57.14	86.89	*(76.04)*	**72.02**
CNG	0	2	PAN12	14.29	80.33	*(57.29)*	**47.31**
Stamatatos-CNG	2	2	PAN12	14.29	78.69	*(60.42)*	**46.49**
Stamatatos-CNG-CN	0	2	CC04	*(42.86)*	52.46	18.75	**35.61**

Table 1: Evaluation Results.

and that there exists a certain number (3) of general grammar rules for English which are used by *all* authors. I.e. those rules cannot by used to infer information about individual authors (e.g. every sentence starts with `[S-...]`).

All other metrics led to worse results, which may also be a result of the fact that only the Sentence-SPI metric makes use of the additional parameters t_c and t_o. Future work should also investigate on integrating these parameters also in other metrics. Moreover, results are better using the PAN12 data set for optimization, which may be because this set is the most hetergeneous one: The Federalist Papers contain only political essays written some time ago, and the CC04 set only uses literary texts written by four authors.

5 Related Work

Successful current approaches often are based on or include character n-grams (e.g. (Hirst and Feiguina, 2007; Stamatatos, 2009)). Several studies have shown that n-grams represent a significant feature to identify authors, whereby the major benefits are the language independency as well as the easy computation. As a variation, word n-grams are used in (Balaguer, 2009) to detect plagiarism in text documents.

Using individual features, machine learning algorithms are often applied to learn from author profiles and to predict unlabeled documents. Among methods that are utilized in authorship attribution as well as the related problem classes like text categorization or intrinsic plagiarism detection are support vector machines (e.g. (Sanderson and Guenter, 2006; Diederich et al., 2000)), neural networks (e.g. (Tweedie et al., 1996)), naive bayes classifiers (e.g. (McCallum and Nigam, 1998)) or decision trees (e.g. (Ö. Uzuner et. al, 2005)).

Another interesting approach used in authorship attribution that tries to detect the writing style of authors by analyzing the occurences and variations of spelling errors is proposed in (Koppel and Schler, 2003). It is based on the assumption that authors tend to make similar spelling and/or grammar errors and therefore uses this information to attribute authors to unseen text documents.

Approaches in the field of genre categorization also use NLP tools to analyze documents based on syntactic annotations (Stamatatos et al., 2000). Lexicalized tree-adjoining-grammars (LTAG) are poposed in (Joshi and Schabes, 1997) as a ruleset to construct and analyze grammar syntax by using partial subtrees.

6 Conclusion and Future Work

In this paper we propose a new feature to enhance modern authorship attribution algorithms by utilizing the grammar syntax of authors. To distinguish between authors, syntax trees of sentences are calculated which are split into parts by using pq-grams. The set of pq-grams is then stored in an author profile that is used to assign unseen documents to known authors.

The algorithm has been optimized and evaluated using three different data sets, resulting in an overall attribution rate of 72%. As the work in this paper solely used the grammar feature and completely ignores information like the vocabulary richness or n-grams, the evaluation results are promising. Future work should therefore concentrate on integrating other well-known and good-working features as well as considering common machine-learning techniques like support vector machines or decision trees to predict authors based on pq-gram features. Furthermore, the optimization parameters currently only applied on the similiarity score should also be integrated with the distance metrics as they led to the best results. Research should finally also be done on the applicability to other languages, especially as syntactically more complex languages like German or French may lead to better results due to the higher amount of grammar rules, making the writing style of authors more unique.

References

Nikolaus Augsten, Michael Böhlen, and Johann Gamper. 2010. The pq-Gram Distance between Ordered Labeled Trees. *ACM Transactions on Database Systems (TODS)*.

Enrique Vallés Balaguer. 2009. Putting Ourselves in SME's Shoes: Automatic Detection of Plagiarism by the WCopyFind tool. In *Proceedings of the SEPLN'09 Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse*, pages 34–35.

Joachim Diederich, Jörg Kindermann, Edda Leopold, and Gerhard Paass. 2000. Authorship attribution with support vector machines. *APPLIED INTELLIGENCE*, 19:2003.

Georgia Frantzeskou, Efstathios Stamatatos, Stefanos Gritzalis, and Sokratis Katsikas. 2006. Effective identification of source code authors using byte-level information. In *Proceedings of the 28th international conference on Software engineering*, pages 893–896. ACM.

Graeme Hirst and Ol'ga Feiguina. 2007. Bigrams of syntactic labels for authorship discrimination of short texts. *Literary and Linguistic Computing*, 22(4):405–417.

Aravind K. Joshi and Yves Schabes. 1997. Tree-adjoining grammars. In *Handbook of formal languages*, pages 69–123. Springer.

Patrick Juola. 2006. Authorship attribution. *Foundations and Trends in Information Retrieval*, 1(3):233–334.

Patrick Juola. 2012. An overview of the traditional authorship attribution subtask. In *CLEF (Online Working Notes/Labs/Workshop)*.

Vlado Kešelj, Fuchun Peng, Nick Cercone, and Calvin Thomas. 2003. N-gram-based author profiles for authorship attribution. In *Proceedings of the conference pacific association for computational linguistics, PACLING*, volume 3, pages 255–264.

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

Moshe Koppel and Jonathan Schler. 2003. Exploiting Stylistic Idiosyncrasies for Authorship Attribution. In *IJCAI'03 Workshop On Computational Approaches To Style Analysis And Synthesis*, pages 69–72.

Mitchell P. Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19:313–330, June.

Andrew McCallum and Kamal Nigam. 1998. A comparison of event models for naive bayes text classification.

F. Mosteller and D. Wallace. 1964. *Inference and Disputed Authorship: The Federalist*. Addison-Wesley.

Smita Nirkhi and RV Dharaskar. 2013. Comparative study of authorship identification techniques for cyber forensics analysis. *International Journal*.

Ö. Uzuner et. al. 2005. Using Syntactic Information to Identify Plagiarism. In *Proc. 2nd Workshop on Building Educational Applications using NLP*.

Conrad Sanderson and Simon Guenter. 2006. Short text authorship attribution via sequence kernels, markov chains and author unmasking: an investigation. In *Proc. of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 482–491, Stroudsburg, PA, USA.

Efstathios Stamatatos, George Kokkinakis, and Nikos Fakotakis. 2000. Automatic text categorization in terms of genre and author. *Comput. Linguist.*, 26:471–495, December.

Efstathios Stamatatos. 2007. Author identification using imbalanced and limited training texts. In *Database and Expert Systems Applications, 2007. DEXA'07. 18th International Workshop on*, pages 237–241. IEEE.

Efstathios Stamatatos. 2009. Intrinsic Plagiarism Detection Using Character n-gram Profiles. In *CLEF (Notebook Papers/Labs/Workshop)*.

Michael Tschuggnall and Günther Specht. 2013a. Countering Plagiarism by Exposing Irregularities in Authors Grammars. In *EISIC, European Intelligence and Security Informatics Conference, Uppsala, Sweden*, pages 15–22.

Michael Tschuggnall and Günther Specht. 2013b. Detecting Plagiarism in Text Documents through Grammar-Analysis of Authors. In *15. GI-Fachtagung Datenbanksysteme für Business, Technologie und Web, Magdeburg, Germany*.

Michael Tschuggnall and Günther Specht. 2013c. Using grammar-profiles to intrinsically expose plagiarism in text documents. In *NLDB*, pages 297–302.

Fiona J. Tweedie, S. Singh, and David I. Holmes. 1996. Neural network applications in stylometry: The federalist papers. *Computers and the Humanities*, 30(1):1–10.

Multi-Domain Sentiment Relevance Classification with Automatic Representation Learning

Christian Scheible
Institut für Maschinelle Sprachverarbeitung
University of Stuttgart
scheibcn@ims.uni-stuttgart.de

Hinrich Schütze
Center for Information
and Language Processing
University of Munich

Abstract

Sentiment relevance (SR) aims at identifying content that does not contribute to sentiment analysis. Previously, automatic SR classification has been studied in a limited scope, using a single domain and feature augmentation techniques that require large hand-crafted databases. In this paper, we present experiments on SR classification with automatically learned feature representations on multiple domains. We show that a combination of transfer learning and in-task supervision using features learned unsupervisedly by the stacked denoising autoencoder significantly outperforms a bag-of-words baseline for in-domain and cross-domain classification.

1 Introduction

Many approaches to sentiment analysis rely on term-based clues to detect the polarity of sentences or documents, using the bag-of-words (BoW) model (Wang and Manning, 2012). One drawback of this approach is that the polarity of a clue is often treated as fixed, which can be problematic when content is not intended to contribute to the polarity of the entity but contains a term with a known lexical non-neutral polarity.

For example, movie reviews often have plot summaries which contain subjective descriptions, e.g., "April loves her new home and friends.", containing "loves", commonly a subjective positive term. Other domains contain different types of nonrelevant content: Music reviews may contain track listings, product reviews on retail platforms contain complaints that do not concern the product, e.g., about shipping and handling. Filtering such nonrelevant content can help to improve sentiment analysis (Pang and Lee, 2004). *Sentiment relevance* (Scheible and Schütze, 2013; Taboada et al., 2009; Täckström and McDonald, 2011) formalizes this distinction: Content that contributes to the overall sentiment of a document is said to be *sentiment relevant* (*SR*), other content is *sentiment nonrelevant* (*SNR*).

The main bottleneck in automatic SR classification is the lack of annotated data. On the sentence level, it has been attempted for the movie review domain (Scheible and Schütze, 2013) on a manually annotated dataset that covers around 3,500 sentences. The sentiment analysis data by Täckström and McDonald (2011) contains SR annotations for five product review domains, four of which have fewer than 1,000 annotated examples.

As the amount of labeled data is low, we adopt *transfer learning* (*TL*, (Thrun, 1995)), which has been used before for SR classification. In this setup, we train a classifier on a different task, using subjectivity-labeled data – for which a large number of annotated examples is available – and apply it for SR classification. To enable knowledge transfer between the tasks, feature space augmentation has been proposed. For this purpose, we employ automatic representation learning, using the stacked denoising autoencoder (SDA, (Vincent et al., 2010)) which has been applied successfully to other domain adaptation problems such as cross-domain sentiment analysis (Glorot et al., 2011).

In this paper, we present experiments on both multi-domain and cross-domain SR classification. We show that compared to the in-domain baseline, TL with SDA features increases F_1 by 6.8% on average. We find that domain adaptation using TL with the SDA compensates for strong domain shifts, reducing the average classification transfer loss by 12.7%.

2 Stacked Denoising Autoencoders

The *stacked denoising autoencoder* (*SDA*, (Vincent et al., 2010)) is a neural network (NN) model for unsupervised feature representation learning.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 200–204,
Gothenburg, Sweden, April 26-30 2014.

An *autoencoder* takes an input vector $\mathbf{x}$, uses an NN layer with a (possibly) nonlinear activation function to generate a hidden feature representation $\mathbf{h}$. A second NN layer reconstructs $\mathbf{x}$ at the output, minimizing the error.

Denoising autoencoders reconstruct $\mathbf{x}$ from a corrupted version of the input, $\tilde{\mathbf{x}}$. As the model learns to be robust to noise, the representations are expected to generalize better. For discrete data, masking noise is a natural choice, where each input unit is randomly set to 0 with probability p.

Autoencoders can be *stacked* by using the $\mathbf{h_i}$ produced by the i^{th} autoencoder as the input to the $(i+1)^{\text{th}}$ one, yielding the representation $\mathbf{h_{i+1}}$. The $\mathbf{h}$ of the topmost autoencoder is the final representation output by the SDA. We let k-SDA denote a stack of k denoising autoencoders.

Chen et al. (2012) introduced a marginalized closed-form version, the mSDA. We opt for this version as it is faster to train and allows us to use the full feature space which would be inefficient with iterative backpropagation training.

3 Task and Experimental Setup

The task in this paper is multi- and cross-domain SR classification. Two aspects motivate our work: First, we need to address the sparse data situation. Second, we are interested in how cross-domain effects influence SR classification. We classify SR in three different setups: in-domain (ID), in which we take the training and test data from the same domain; domain adaptation (DA), where training and test data are from different domains; and transfer learning (TL), where we use a much larger amount of data from a different but related task. To improve the generalization capabilities of the models, we use representations learned by the SDA. We will next describe our classification setup in more detail.

Data We use the following datasets for our experiments. Table 1 shows statistics on the datasets.

CINEMA: The movie SR data (CINEMA) by Scheible and Schütze (2013) contains SR-annotated sentences for the movie review domain. Ambiguous sentences are marked as *unknown*; we exclude them.

PRODUCTS: The multi-domain product data (PRODUCTS) by Täckström and McDonald (2011) contains labeled sentences from five Amazon.com product review domains: BOOKS, DVDS, electronics (EL), MUSIC, and video games (VG). This dataset differs from CINEMA firstly in the product domains (except obviously for DVDS which also covers movies). Secondly, the data was collected from a retail site, which introduces further facets of sentiment nonrelevance, as discussed above. Thirdly, the annotation style has no *unknown* category: ambiguous examples are marked as SR.

Dataset	#doc	#sent	#SR	#SNR
CINEMA	125	3,487	2,759	728
PRODUCTS	294	3,836	2,689	1,147
–BOOKS	59	739	424	315
–DVDS	59	799	524	275
–ELECTRONICS	57	628	491	137
–MUSIC	59	638	448	190
–VIDEOGAMES	60	1032	802	230
P&L	–	10,000	5,000	5,000
UNLAB	7,500	68,927	–	–

Table 1: Dataset statistics

P&L: The subjectivity data (P&L) by Pang and Lee (2004) serves as our cross-task training data for transfer learning. The dataset was heuristically created for subjectivity detection on the movie domain by sampling snippets from Rotten Tomatoes as subjective and sentences from IMDb plot summaries as objective examples.

UNLAB: To improve generalization on PRODUCTS, we use additional unlabeled sentences (UNLAB) for SDA training. We extract the sentences of 1,500 randomly selected documents for each of the five domains from the Amazon.com review data by Jindal and Liu (2008).

SDA setup We train the SDA with 10,000 hidden units and tanh nonlinearity on the BoW features of all available data as the input. We optimize the noise level p with 2-fold cross-validation on the in-domain training folds.

Classification setup We perform SR classification with a linear support vector machine (SVM) using LIBLINEAR (Chang and Lin, 2011). We perform 2-fold cross-validation for all training data but P&L. We report overall macro-averaged F_1 over both folds. The feature representation for the SVM is either bag of words (BoW) or the k-SDA output. Unlike Chen et al. (2012), we do not use concatenations of BoW and SDA vectors as we found them to perform worse.

Evaluation As class distributions are heavily skewed, we use *macro-averaged* $F_1(s,t)$ (training on s and evaluating on t) as the basic evaluation measure. We evaluate DA with *transfer loss*, the difference in F_1 of a classifier CL with

	Features	Setup	CINEMA	BOOKS	DVDS	EL	MUSIC	VG	∅
1	Majority BL	–	39.6	28.9	32.6	39.2	35.1	39.0	35.7
2	BoW	ID	74.0	57.5	49.8	55.1	55.5	55.0	58.4
3	1-SDA	ID	73.6	55.3	48.4	43.8	41.8	44.1	52.6
4	2-SDA	ID	76.0	54.5	52.5	43.9	41.2	46.7	53.6
5	BoW	TL	71.5	60.7	60.2	50.3	55.1	53.2	59.6
6	1-SDA	TL	73.3	**62.9**	60.6	**59.0**	**59.9**	57.0	63.1
7	2-SDA	TL	76.2	**62.9**	**65.8**	**59.7**	**59.9**	**60.5**	**64.9**
8	BoW	ID+TL	76.6	**63.5**	61.7	52.4	56.7	57.0	62.3
9	1-SDA	ID+TL	**79.0**	**62.7**	62.1	**57.7**	**57.8**	**57.4**	63.9
10	2-SDA	ID+TL	**80.4**	**62.7**	**65.2**	**59.0**	**58.7**	**58.9**	**65.2**

Table 2: Macro-averaged F_1 (%) evaluating on each test domain on both folds. ∅ = row mean. **Bold:** best result in each column and results in that column not significantly different from it.

respect to the in-domain baseline BL: $L(s,t) = F_1^{(\mathrm{BL})}(t,t) - F_1^{(\mathrm{CL})}(s,t)$. L is negative if the classifier surpasses the baseline. As a statistical significance test (indicated by † in the text), we use *approximate randomization* (Noreen, 1989) with 10,000 iterations at $p < 0.05$.

4 Experiments

In-Domain Classification (ID) Table 2 shows macro-averaged F_1 for different SR models. We first turn to fully supervised SR classification with bag-of-words (BoW) features using ID training (line 2). While the results for CINEMA are high, on par with the reported results in related work, they are low for the PRODUCTS data. This is not surprising as the SVM is trained with fewer than 600 examples on each domain. Also, no *unknown* category exists in the latter dataset. While ambiguous examples on CINEMA are annotated as *unknown*, they receive an SR label on PRODUCTS. Thus, many examples are ambiguous and thus difficult to classify. SDA features worsen results significantly† (lines 3–4) on all domains except CINEMA and DVDS due to data sparsity. They are the two most homogeneous domains where plot descriptions make up a large part of the SNR content. On many domains, there is no single prototypical type of SNR which could be learned from a small amount of training data.

Transfer Learning (TL) TL with training on P&L and evaluation on CINEMA/PRODUCTS with BoW features (line 5) performs slightly worse than ID classification, except on BOOKS and DVDS where we see strong improvements. This result is easy to explain: Both BOOKS and DVDS contain SNR descriptions of narratives, which are covered well in P&L. This distinction is less helpful on domains like EL where SNR content is different, so we achieve worse results even with the much larger P&L data.

We find that 1-SDA (line 6) already performs significantly† better than the ID baseline on all domains except CINEMA which has a much larger amount of ID training data available than the other domains (approx. 1700 sentences vs. fewer than 600). Using stacking, 2-SDA (line 7) improves the results on three domains significantly,† and performs on par with the ID classifier on CINEMA. We found that stack depths of $k > 2$ do not significantly† increase performance.

Finally, we try a combination of ID and TL (ID+TL), training on both P&L and the respective ID training fold of CINEMA/PRODUCTS. The results for this experiment are shown in lines 8–10 in Table 2. Comparing BoW models, we beat both ID and TL across all domains (lines 2 and 5). With SDA features, we are able to beat ID for CINEMA. The results on the other domains are comparable to plain TL. This is a promising result, showing that with SDA features, ID+TL performs as well as or better than plain TL. This property could be exploited for domains where labeled data is not available. We will show below that SDA features become important when we apply ID+TL to domain adaptation.

We also conducted experiments using only the 5,000 most frequent features but found that the SDA does not generalize well from this input representation, particularly on EL and MUSIC. This confirms that in SR, rare features make an important contribution (such as named entities in the movie domain).

Domain Adaptation (DA) We now evaluate the task in a DA setting, comparing the ID and ID+TL

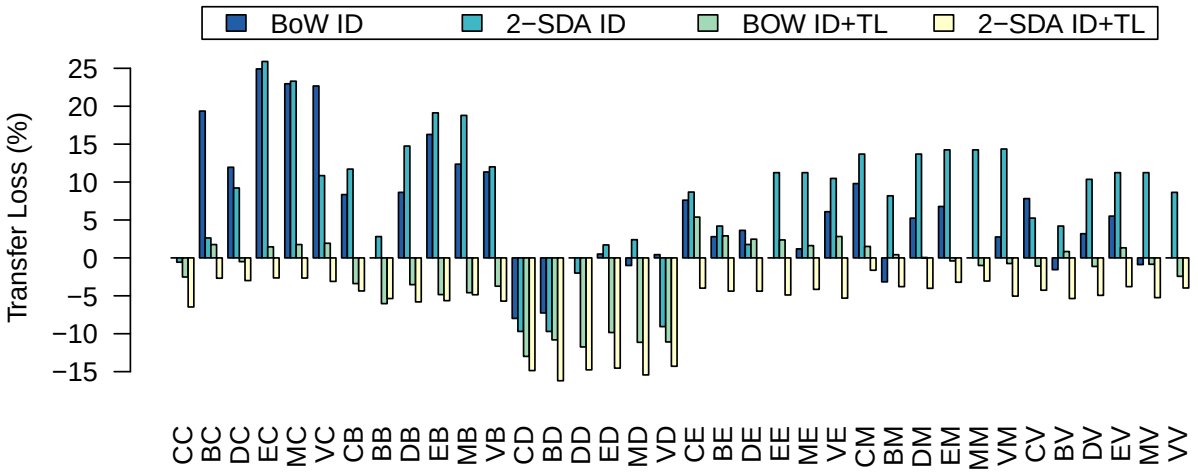


Figure 1: Transfer losses (%) for DA. Training-test pairs grouped by target domain and abbreviated by first letter (e.g., CD: training on CINEMA, evaluating on DVDS). In-domain results shown for comparison to Table 2.

setups with BoW and 2-SDA features. We measure the transfer losses we suffer from training on one domain and evaluating on another (Figure 1). The overall picture is the same as above: ID+TL 2-SDA models perform best. In the baseline BoW ID setup, domain shifts have a strong influence on the results. The combination of out-of-domain and out-of-task data in ID+TL keeps losses uniformly low. 2-SDA features lower almost all losses further. On average, 2-SDA ID+TL reduces transfer loss by 12.7 points compared to the baseline (Table 3). As expected, pairings of thematically strongly related domains (e.g., BOOKS and DVDS) have lower losses in all setups.

The biggest challenge is the strong domain shift between the CINEMA and PRODUCTS domains (concerning mainly the retail aspects). With BoW ID, losses on CINEMA reach up to 25 points, and using CINEMA for training causes high losses for PRODUCTS in most cases. Our key result is that the ID+TL 2-SDA setup successfully compensates for these problems, reducing the losses below 0.

Losses across the PRODUCTS domains are less pronounced. The DVDS baseline classifier has the lowest F_1 (cf. Table 2) and shows the highest improvements in domain adaptation: BoW models of other domains perform better than the in-domain classifier. Analyzing the DVDS model shows overfitting to specific movie terms which occur frequently across each review in the training data. SNR content in movies is mostly concerned with named entity types which cannot easily be learned from BoW representations. Out-of-domain models are less specialized and perform better than in-domain models. TL and SDA increase the coverage of movie terms and provide better generalization, which improves performance further.

	BoW	2-SDA
ID	6.7	8.9
ID+TL	-1.8	-6.0

Table 3: Mean transfer losses (%) for the different training data and feature representation setups. In-domain results not included.

BOOKS is the most challenging domain in all setups. It is particularly heterogeneous, containing both fiction and non-fiction reviews which feature different SNR aspects. Both results illustrate that domain effects depend on how diverse SNR content is within the domain.

Overall, the results show that ID+TL leads to a successful compensation of cross-domain effects. SDA features improve the results significantly† for ID+TL. In particular, we find that the SDA successfully compensates for the strong domain shift between CINEMA and PRODUCTS.

5 Conclusion

We presented experiments on multi- and cross-domain sentiment relevance classification. We showed that transfer learning (TL) using stacked denoising autoencoder (SDA) representations significantly increases performance by 6.8% F_1 for in-domain classification. Moreover, the average transfer loss in domain adaptation is reduced by 12.7 percentage points where the SDA features compensate for strong domain shifts.

References

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

Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on Machine Learning (ICML)*.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, pages 513–520.

Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the International Conference on Web Search and Web Data Mining (WSDM)*, pages 219–230.

Eric W. Noreen. 1989. *Computer Intensive Methods for Hypothesis Testing: An Introduction*. Wiley.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Meeting of the Association for Computational Linguistics (ACL)*, pages 271–278.

Christian Scheible and Hinrich Schütze. 2013. Sentiment relevance. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 954–963.

Maite Taboada, Julian Brooke, and Manfred Stede. 2009. Genre-based paragraph classification for sentiment analysis. In *Proceedings of the SIGDIAL 2009 Conference*, pages 62–70.

Oscar Täckström and Ryan McDonald. 2011. Semi-supervised latent variable models for sentence-level sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 569–574.

Sebastian Thrun. 1995. Is learning the n-th thing any easier than learning the first? In *Proceedings of Advances in Neural Information Processing Systems 8 (NIPS)*, pages 640–646.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *The Journal of Machine Learning Research (JMLR)*, 11:3371–3408.

Sida Wang and Christopher Manning. 2012. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 90–94.

A New Entity Salience Task with Millions of Training Examples

Jesse Dunietz
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
jdunietz@cs.cmu.edu

Dan Gillick
Google Research
1600 Amphitheatre Parkway
Mountain View, CA 94043, USA
dgillick@google.com

Abstract

Although many NLP systems are moving toward entity-based processing, most still identify important phrases using classical keyword-based approaches. To bridge this gap, we introduce the task of *entity salience*: assigning a relevance score to each entity in a document. We demonstrate how a labeled corpus for the task can be automatically generated from a corpus of documents and accompanying abstracts. We then show how a classifier with features derived from a standard NLP pipeline outperforms a strong baseline by 34%. Finally, we outline initial experiments on further improving accuracy by leveraging background knowledge about the relationships between entities.

1 Introduction

Information retrieval, summarization, and online advertising rely on identifying the most important words and phrases in web documents. While traditional techniques treat documents as collections of keywords, many NLP systems are shifting toward understanding documents in terms of entities. Accordingly, we need new algorithms to determine the prominence – the *salience* – of each entity in the document.

Toward this end, we describe three primary contributions. First, we show how a labeled corpus for this task can be automatically constructed from a corpus of documents with accompanying abstracts. We also demonstrate the validity of the corpus with a manual annotation study. Second, we train an entity salience model using features derived from a coreference resolution system. This model significantly outperforms a baseline model based on sentence position. Third, we suggest how our model can be improved by leveraging background information about the entities and their relationships – information not specifically provided in the document in question.

Our notion of salience is similar to that of Boguraev and Kenney (1997): "discourse objects with high salience are the focus of attention", inspired by earlier work on Centering Theory (Walker et al., 1998). Here we take a more empirical approach: salient entities are those that human readers deem most relevant to the document.

The entity salience task in particular is briefly alluded to by Cornolti et al. (2013), and addressed in the context of Twitter messages by Meij et. al (2012). It is also similar in spirit to the much more common keyword extraction task (Tomokiyo and Hurst, 2003; Hulth, 2003).

2 Generating an entity salience corpus

Rather than manually annotating a corpus, we automatically generate salience labels for an existing corpus of document/abstract pairs. We derive the labels using the assumption that the salient entities will be mentioned in the abstract, so we identify and align the entities in each text.

Given a document and abstract, we run a standard NLP pipeline on both. This includes a POS tagger and dependency parser, comparable in accuracy to the current Stanford dependency parser (Klein and Manning, 2003); an NP extractor that uses POS tags and dependency edges to identify a set of entity mentions; a coreference resolver, comparable to that of Haghighi and Klein, (2009) for clustering mentions; and an entity resolver that links entities to Freebase profiles. The entity resolver is described in detail by Lao, et al. (2012).

We then apply a simple heuristic to align the entities in the abstract and document: Let M_E be the set of mentions of an entity E that are proper names. An entity E_A from the abstract aligns to an entity E_D from the document if the syntactic head token of some mention in M_{E_A} matches the head token of some mention in M_{E_D}. If E_A aligns with more than one document entity, we align it with the document entity that appears earliest.

In general, aligning an abstract to its source document is difficult (Daumé III and Marcu, 2005).

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 205–209,
Gothenburg, Sweden, April 26-30 2014.

We avoid most of this complexity by aligning only entities with at least one proper-name mention, for which there is little ambiguity. Generic mentions like *CEO* or *state* are often more ambiguous, so resolving them would be closer to the difficult problem of word sense disambiguation.

Once we have entity alignments, we assume that a document entity is salient only if it has been aligned to some abstract entity. Ideally, we would like to induce a salience ranking over entities. Given the limitations of short abstracts, however, we settle for binary classification, which still captures enough salience information to be useful.

2.1 The New York Times corpus

Our corpus of document/abstract pairs is the annotated New York Times corpus (Sandhaus, 2008). It includes 1.8 million articles published between January 1987 and June 2007; some 650,000 include a summary written by one of the newspaper's library scientists. We selected a subset of the summarized articles from 2003-2007 by filtering out articles and summaries that were very short or very long, as well as several special article types (e.g., corrections and letters to the editor).

Our full labeled dataset includes 110,639 documents with 2,229,728 labeled entities; about 14% are marked as salient. For comparison, the average summary is about 6% of the length (in tokens) of the associated article. We use the 9,719 documents from 2007 as test data and the rest as training.

2.2 Validating salience via manual evaluation

To validate our alignment method for inferring entity salience, we conducted a manual evaluation. Two expert linguists discussed the task and generated a rubric, giving them a chance to calibrate their scores. They then independently annotated all detected entities in 50 random documents from our corpus (a total of 744 entities), without reading the accompanying abstracts. Each entity was assigned a salience score in $\{1, 2, 3, 4\}$, where 1 is most salient. We then thresholded the annotators' scores as salient/non-salient for comparison to the binary NYT labels.

Table 1 summarizes the agreement results, measured by Cohen's kappa. The experts' agreement is probably best described as *moderate*,[1] indicating that this is a difficult, subjective task, though deciding on the most salient entities (with score 1) is easier. Even without calibrating to the induced NYT salience scores, the expert vs. NYT agreement is close enough to the inter-expert agreement to convince us that our induced labels are a reasonable if somewhat noisy proxy for the experts' definition of salience.

Comparison	$\kappa_{\{1,2\}}$	$\kappa_{\{1\}}$
A1 vs. A2	0.56	0.69
A1 vs. NYT	0.36	0.48
A2 vs. NYT	0.39	0.35
A1 & A2 vs. NYT	0.43	0.38

Table 1: Annotator agreement for entity salience as a binary classification. A1 and A2 are expert annotators; NYT represents the induced labels. The first κ column assumes annotator scores $\{1, 2\}$ are salient and $\{3, 4\}$ are non-salient, while the second κ column assumes only scores of 1 are salient.

3 Salience classification

We built a regularized binary logistic regression model to predict the probability that an entity is salient. To simplify feature selection and to add some further regularization, we used feature hashing (Ganchev and Dredze, 2008) to randomly map each feature string to an integer in $[1, 100000]$; larger alphabet sizes yielded no improvement. The model was trained with L-BGFS.

3.1 Positional baseline

For news documents, it is well known that sentence position is a very strong indicator for relevance. Thus, our baseline is a system that identifies an entity as salient if it is mentioned in the first sentence of the document. (Including the next few sentences did not significantly change the score.)

3.2 Model features

Table 2 describes our feature classes; each individual feature in the model is a binary indicator. Count features are bucketed by applying the function $f(x) = \text{round}(\log(k(x + 1)))$, where k can be used to control the number of buckets. We simply set $k = 10$ in all cases.

3.3 Experimental results

Table 3 shows experimental results on our test set. Each experiment uses a classification threshold of 0.3 to determine salience, which in each case is very close to the threshold that maximizes F_1. For comparison, a classifier that always predicts the majority class, non-salient, has $F_1 = 23.9$ (for the *salient* class).

[1] For comparison, word sense disambiguation tasks have reported agreement as low as $\kappa = 0.3$ (Yong and Foo, 1999).

Feature name	Description
`1st-loc`	Index of the sentence in which the first mention of the entity appears.
`head-count`	Number of times the head word of the entity's first mention appears.
`mentions`	Conjuction of the numbers of named (*Barack Obama*), nominal (*president*), pronominal (*he*), and total mentions of the entity.
`headline`	POS tag of each word that appears in at least one mention and also in the headline.
`head-lex`	Lowercased head word of the first mention.

Table 2: The feature classes used by the classifier.

#	Description	P	R	F_1
1	Positional baseline	59.5	37.8	46.2
2	`head-count`	37.3	54.7	44.4
3	`mentions`	57.2	51.3	54.1
4	`1st-loc`	46.1	60.2	52.2
5	+ `head-count`	52.6	63.4	57.5
6	+ `mentions`	59.3	61.3	60.3
7	+ `headline`	59.1	61.9	60.5
8	+ `head-lex`	59.7	63.6	61.6
9	+ `centrality`	60.5	63.5	62.0

Table 3: Test set (P)recision, (R)ecall, and (F) measure of the *salient* class for some combinations of features listed in Table 2. The `centrality` feature is discussed in Section 4.

Lines 2 and 3 serve as a comparison between traditional keyword counts and the mention counts derived from our coreference resolution system. Named, nominal, and pronominal mention counts clearly add significant information despite coreference errors. Lines 4-8 show results when our model features are incrementally added. Each feature raises accuracy, and together our simple set of features improves on the baseline by 34%.

4 Entity centrality

All the features described above use only information available within the document. But articles are written with the assumption that the reader knows something about at least some of the entities involved. Inspired by results using Wikipedia to improve keyword extraction tasks (Mihalcea and Csomai, 2007; Xu et al., 2010), we experimented with a simple method for including background knowledge about each entity: an adaptation of PageRank (Page et al., 1999) to a graph of connected entities, in the spirit of Erkan and Radev's work (2004) on summarization.

Consider, for example, an article about a recent congressional budget debate. Although House Speaker John Boehner may be mentioned just once, we know he is likely salient because he is closely related to other entities in the article, such as Congress, the Republican Party, and Barack Obama. On the other hand, the Federal Emergency Management Agency may be mentioned repeatedly because it happened to host a major presidential speech, but it is less related to the story's key figures and less central to the article's point.

Our intuition about these relationships, mostly not explicit in the document, can be formalized in a local PageRank computation on the entity graph.

4.1 PageRank for computing centrality

In the weighted version of the PageRank algorithm (Xing and Ghorbani, 2004), a web link is considered a weighted vote by the containing page for the landing page – a directed edge in a graph where each node is a webpage. In place of the web graph, we consider the graph of Freebase entities that appear in the document. The nodes are the entities, and a directed edge from E_1 to E_2 represents $P(E_2|E_1)$, the probability of observing E_2 in a document given that we have observed E_1. We estimate $P(E_2|E_1)$ by counting the number of training documents in which E_1 and E_2 co-occur and normalizing by the number of training documents in which E_1 occurs.

The nodes' initial PageRank values act as a prior, where the uniform distribution, used in the classic PageRank algorithm, indicates a lack of prior knowledge. Since we have some prior signal about salience, we initialize the node values to the normalized mention counts of the entities in the document. We use a damping factor d, allowing random jumps between nodes with probability $1 - d$, with the standard value $d = 0.85$.

We implemented the iterative version of weighted PageRank, which tends to converge in under 10 iterations. The `centrality` features in Table 3 are indicators for the rank orders of the converged entity scores. The improvement from adding centrality features is small but statistically significant at $p \leq 0.001$.

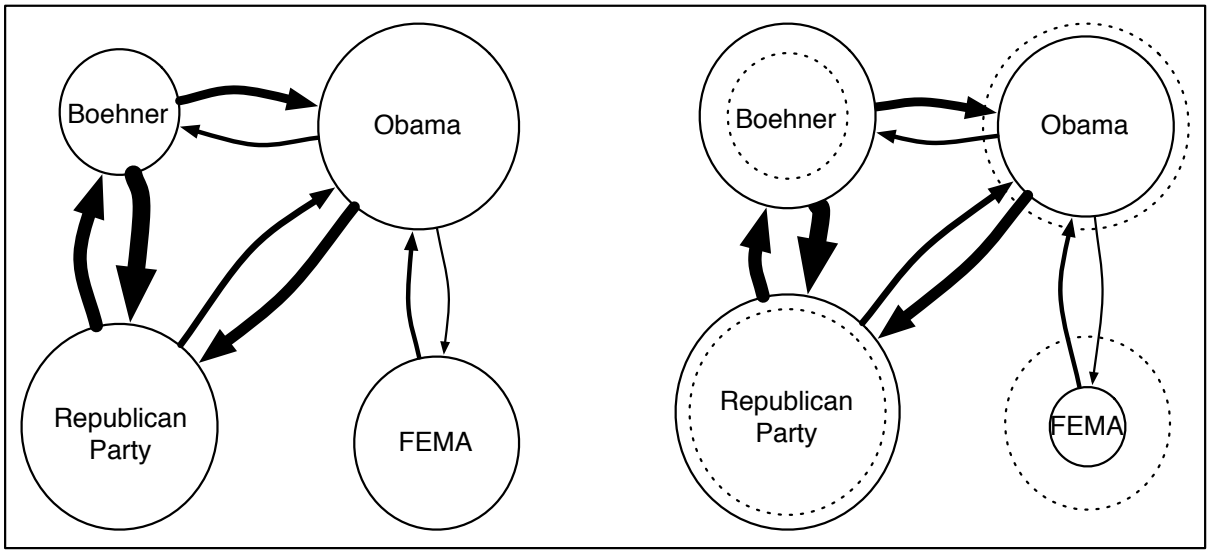


Figure 1: A graphical representation of the centrality computation on a toy example. Circle size and arrow thickness represent node value and edge weight, respectively. The initial node values, based on mention count, are shown on the left. The final node values are on the right; dotted circles show the initial sizes for comparison. Edge weights remain constant.

4.2 Discussion

We experimented with a number of variations on this algorithm, but none gave much meaningful improvement. In particular, we tried to include the neighbors of all entities to increase the size of the graph, with the values of neighbor entities not in the document initialized to some small value k. We set a minimum co-occurrence count for an edge to be included, varying it from 1 to 100 (where 1 results in very large graphs). We also tried using Freebase relations between entities (rather than raw co-occurrence counts) to determine the set of neighbors. Finally, we experimented with undirected graphs using unnormalized co-occurrence counts.

While the ranked centrality scores look reasonable for most documents, the addition of these features does not produce a substantial improvement. One potential problem is our reliance on the entity resolver. Because the PageRank computation links all of a document's entities, a single resolver error can significantly alter all the centrality scores. Perhaps more importantly, the resolver is incomplete: many tail entities are not included in Freebase.

Still, it seems likely that even with perfect resolution, entity centrality would not significantly improve the accuracy of our model. The `mentions` features are sufficiently powerful that entity centrality seems to add little information to the model beyond what these features already provide.

5 Conclusions

We have demonstrated how a simple alignment of entities in documents with entities in their accompanying abstracts provides salience labels that roughly agree with manual salience annotations. This allows us to create a large corpus – over 100,000 labeled documents with over 2 million labeled entities – that we use to train a classifier for predicting entity salience.

Our experiments show that features derived from a coreference system are more robust than simple word count features typical of a keyword extraction system. These features combine nicely with positional features (and a few others) to give a large improvement over a first-sentence baseline.

There is likely significant room for improvement, especially by leveraging background information about the entities, and we have presented some initial experiments in that direction. Perhaps features more directly linked to Wikipedia, as in related work on keyword extraction, can provide more focused background information.

We believe entity salience is an important task with many applications. To facilitate further research, our automatically generated salience annotations, along with resolved entity ids, for the subset of the NYT corpus discussed in this paper are available here:
https://code.google.com/p/nyt-salience/

References

Branimir Boguraev and Christopher Kennedy. 1997. Salience-based content characterisation of text documents. In *Proceedings of the ACL*, volume 97, pages 2–9.

Marco Cornolti, Paolo Ferragina, and Massimiliano Ciaramita. 2013. A framework for benchmarking entity-annotation systems. In *Proceedings of the 22nd international conference on World Wide Web*, pages 249–260.

Hal Daumé III and Daniel Marcu. 2005. Induction of word and phrase alignments for automatic document summarization. *Computational Linguistics*, 31(4):505–530.

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR)*, 22(1):457–479.

Kuzman Ganchev and Mark Dredze. 2008. Small statistical models by random feature mixing. In *Proceedings of the ACL08 HLT Workshop on Mobile Language Processing*, pages 19–20.

Aria Haghighi and Dan Klein. 2009. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*, pages 1152–1161. Association for Computational Linguistics.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223.

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

Ni Lao, Amarnag Subramanya, Fernando Pereira, and William W Cohen. 2012. Reading the web with learned syntactic-semantic inference rules. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1017–1026. Association for Computational Linguistics.

Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding semantics to microblog posts. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pages 563–572. ACM.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242.

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The PageRank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab.

Evan Sandhaus. 2008. The New York Times annotated corpus. *Linguistic Data Consortium, Philadelphia*, 6(12):e26752.

Takashi Tomokiyo and Matthew Hurst. 2003. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment-Volume 18*, pages 33–40.

Marilyn A Walker, Aravind Krishna Joshi, and Ellen Friedman Prince. 1998. *Centering theory in discourse*. Oxford University Press.

Wenpu Xing and Ali Ghorbani. 2004. Weighted pagerank algorithm. In *Communication Networks and Services Research*, pages 305–314. IEEE.

Songhua Xu, Shaohui Yang, and Francis Chi-Moon Lau. 2010. Keyword extraction and headline generation using novel word features. In *AAAI*.

Chung Yong and Shou King Foo. 1999. A case study on inter-annotator agreement for word sense disambiguation.

Finding middle ground? Multi-objective Natural Language Generation from time-series data

Dimitra Gkatzia, Helen Hastie, and Oliver Lemon
School of Mathematical and Computer Sciences, Heriot-Watt University, Edinburgh
{dg106, h.hastie, o.lemon}@hw.ac.uk

Abstract

A Natural Language Generation (NLG) system is able to generate text from non-linguistic data, ideally personalising the content to a user's specific needs. In some cases, however, there are multiple stakeholders with their own individual goals, needs and preferences. In this paper, we explore the feasibility of combining the preferences of two different user groups, lecturers and students, when generating summaries in the context of student feedback generation. The preferences of each user group are modelled as a multivariate optimisation function, therefore the task of generation is seen as a multi-objective (MO) optimisation task, where the two functions are combined into one. This initial study shows that treating the preferences of each user group equally smooths the weights of the MO function, in a way that preferred content of the user groups is not presented in the generated summary.

1 Introduction

Summarisation of time-series data refers to the task of automatically generating summaries from attributes whose values change over time. Content selection is the task of choosing what to say, i.e. what information to be included in a report (Reiter and Dale, 2000). Here, we consider the task of automatically generating feedback summaries for students describing their performance during the lab of a computer science module over the semester. This work is motivated by the fact that different user groups have different preferences of the content that should be conveyed in a summary, as shown by Gkatzia et al. (2013).

Various factors can influence students' learning, such as difficulty of the material (Person et al., 1995), workload (Craig et al., 2004), attendance in lectures (Ames, 1992) etc. These factors change over time and can be interdependent. The different stakeholders (i.e. lecturers and students) have different perceptions regarding what constitutes good feedback. Therefore, when generating feedback, we should take into account all preferences in order to be able to produce feedback summaries that are acceptable by both user groups.

Stakeholders often have conflicting goals, needs and preferences, for example managers with employees or doctors with patients and relatives. In our data, for instance, lecturers tend to comment on the hours that a student studied, whereas the students disprefer this content. Generating the same summary for both groups allows for meaningful further discussion with common ground.

Previous work on NLG systems that address more than one user group use different versions of a system for each different user group (Gatt et al., 2009) or make use of User Models (Janarthanam and Lemon, 2010; Thompson et al., 2004; Zukerman and Litman, 2001). Here, we explore a method that adapts to both expert preferences and users simultaneously (i.e. lecturer and students preferences), by applying Multi-Objective optimisation (MOO). MOO can be applied to situations where optimal decisions are sought in the presence of trade-offs between conflicting objectives (Chankong and Haimes, 1983). We explore whether balancing the preferences of two user groups can result in an adaptive system that is acceptable by all users. At the same time, the programming effort is reduced as only one system needs to be developed. Moreover, by pooling all available data together, there is less need for an extensive data collection.

In the next section, we present three systems: one tuned for lecturers, one for students, and one that attempts to find middle ground. In Section 3, we describe an evaluation of these three systems and in Section 4 we discuss the results. Finally, in

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 210–214,
Gothenburg, Sweden, April 26-30 2014.

Section 5, directions for future work are discussed.

2 Methodology

Reinforcement Learning (RL) is a machine learning technique that defines how an agent learns to take optimal sequences of actions so as to maximize a cumulative reward (Sutton and Barto, 1998). Here we extend the framework proposed by Gkatzia et al. (2013) whereby the content selection is seen as a Markov Decision problem and the goal of the agent is to learn to take the sequence of actions that leads to optimal content selection. A Temporal Difference learning method (Sutton and Barto, 1998) was used to train an agent for content selection. Firstly, we will describe the data in general. Secondly, we refer to the RL system that adapts to lecturers' preferences as described by Gkatzia et al. (2013). Thirdly, we will describe how we collected data and developed a methodology that adapts to students' preferences and finally how we combined the knowledge of both steps to develop an MO system. The three systems (Lecturer-adapted, Student-adapted, MO) share the same architecture but the difference lies in the reward functions used for training.

2.1 The Data

For this study, the dataset described by Gkatzia et al. (2013) was used. Table 1 shows an example of this dataset that describes a student's learning habits and a corresponding feedback summary provided by a lecturer. The dataset is composed of 37 similar instances. Each instance consists of time-series information about the student's learning routine and the selected templates that lecturers used to provide feedback to this student. A template is a quadruple consisting of an *id*, a *factor* (Table 1), a *reference type* (trend, weeks, average, other) and surface text. For instance, a template can be (1, marks, trend, 'Your marks were <trend>over the semester'). The lexical choice for <trend>(i.e. increasing or decreasing) depends on the values of time-series data. There is a direct mapping between the values of factor and reference type and the surface text. The time-series attributes are listed in Table 1 (bottom left).

2.2 Time-series summarisation systems

Actions and states: The state consists of the time-series data and the selected templates. In order to explore the state space the agent selects a time-series attribute (e.g. marks, deadlines etc.) and then decides whether to talk about it or not. The states and actions are similar for all systems.

Lecturer-adapted reward function
The reward function is derived from analysis with linear regression of the provided dataset and is the following cumulative multivariate function:

$$Reward_{LECT} = a + \sum_{i=1}^{n} b_i * x_i + c * length$$

where $X = \{x_1, x_2, ..., x_n\}$ is the vector of combinations of the data trends observed in the time-series data and a particular reference type of the factor. The value of x_i is given by the function:

$$x_i = \begin{cases} 1, & \text{if the combination of a factor trend and a particular reference type is included in the feedback} \\ 0, & \text{if not.} \end{cases}$$

The coefficients represent the preference level of a factor to be selected and how to be conveyed in the summary. Important factors are associated with high positive coefficients and the unimportant ones with negative coefficients. In the training phase, the agent selects a factor and then decides whether to talk about it or not. If it decides to refer to a factor, the selection of the template is performed deterministically, i.e. it selects the template that results in higher reward. Length represents the number of factors selected for generation.

Student-adapted reward function
The Student-adapted system uses the same RL algorithm as the Lecturer-adapted one. The difference lies in the reward function. The reward function used for training is of a similar style as the Lecturer-adapted reward function. This function was derived by manipulating the student ratings in a previous experiment and estimating the weights using linear regression in a similar way as Walker et al. (1997) and Rieser et al. (2010).

Multi-objective function
The function used for the multi-objective method is derived by weighting the sum of the individual reward functions.

$$R_{MO} = 0.5 * R_{LECT} + 0.5 * R_{STUDENT}$$

To reduce the confounding variables, we kept the ordering of content in all systems the same.

3 Evaluation

The output of the above-mentioned three systems were evaluated both in simulation and with real

Raw Data

factors	week 2	week 3	...	week 10
marks	5	4	...	5
hours_studied	1	2	...	3
...	...	...	...	...

Trends from Data

factors	factor trend
(1) marks	trend_other
(2) hours_studied	trend_increasing
(3) understandability	trend_decreasing
(4) difficulty	trend_decreasing
(5) deadlines	trend_increasing
(6) health_issues	trend_other
(7) personal_issues	trend_decreasing
(8) lectures_attended	trend_other
(9) revision	trend_decreasing

Summary

Your overall performance **was excellent** during the semester. Keep up the good work and maybe try some more challenging exercises. Your attendance was **varying** over the semester. Have a think about how to use time in lectures to improve your understanding of the material. You spent **2 hours studying the lecture material on average**. You should dedicate more time to study. You seem to find the material **easier to understand compared to the beginning of the semester**. Keep up the good work! You revised **part of** the learning material. Have a think about whether revising has improved your performance.

Table 1: Top left: example of the time-series raw data for feedback generation. Bottom left: example of described trends. Right box: a target summary generated by an expert (bold signifies the chosen content).

users. Example summaries of all systems are presented in Table 2.

3.1 Evaluation in Simulation

26 summaries were produced by each system. The output of each system was evaluated with the three reward functions. Table 3 shows the results.

As expected, all systems score highly when evaluated with the reward function for which they were trained, with the second highest reward scored from the MO function. Table 2 illustrates this with the MO Policy clearly between the other two policies. Moreover, the MO function reduces the variability between summaries as is also reflected in the standard deviation given in Table 3.

We used BLEU (4-grams) (Papineni et al., 2002) to measure the similarities between the feedback summaries generated by the three systems. BLEU score is between 0-1 with values closer to 1 indicating texts are more similar. Our results demonstrate that the summaries generated by the three systems are quite different (BLEU score between 0.33 and 0.36). This shows that the framework presented here is capable of producing quite different summaries based on the various reward functions.

3.2 Evaluation with real users

The goal of the evaluation is to determine whether the end-user can pick up on the above-mentioned differences in the feedback and rank them according to their preferences. The output of the three systems was ranked by 19 lecturers and 48 first-year Computer Science students. Time-series data of three students were presented on graphs to each participant. They were also shown 3 feedback summaries and they were asked to rank them in terms of preference.

As we can see from Table 4, the two user groups significantly preferred the output of the system which was trained for their preferences (Mann-Whitney U test, $p < 0.05$). Interestingly, lecturers found both the outputs produced by the Lecturer-adapted system and the Student-adapted system significantly preferable ($p < 0.05$) to the output produced by the MO system. In contrast, students significantly preferred the output generated by the Student-adapted system over the other two. Finally, both user groups rated the MO system 3rd, but there is not a significant difference between the student ratings for the MO system and the Lecturer-adapted system.

4 Discussion

It is interesting to examine the weights derived from the multiple-linear regression to determine the preferences of the different user groups. For instance, lecturers' most preferred content is hours_studied, therefore the reward function gives high scores to summaries that mention the hours

Lecturer-adapted	Student-adapted	Multi-objective
Make sure you **revise the learning material** and try to do the lab exercises again. You **dedicated more time studying** the lecture material in the beginning of the semester compared to the end of the semester. Have a think about what is preventing you from studying. Your **understanding** of the material could be improved. Try going over the teaching material again. You have had **other deadlines** during weeks 5, 6, 8, 9 and 10. You may want to plan your studying and work ahead. You did not face any **health problems** during the semester.	You found the **lab exercises very challenging**. Make sure that you have understood the taught material and don't hesitate to ask for clarification. You dedicated more **time studying the lecture material** in the beginning of the semester compared to the end of the semester. Have a think about what is preventing you from studying. Your **understanding** of the material could be improved. Try going over the teaching material again. **Revising material** during the semester will improve your performance in the lab.	Your **attendance** was varying over the semester. Have a think about how to use time in lectures to improve your **understanding** of the material. You found the lab exercises very challenging. Make sure that you have understood the taught material and don't hesitate to ask for clarification. You **dedicated more time studying** the lecture material in the beginning of the semester compared to the end of the semester. Have a think about what is preventing you from studying. You did not face any **health problems** during the semester. You **revised** part of the learning material. Have a think whether revising has improved your performance.

Table 2: Example outputs from the three different systems (bold signifies the chosen content).

Time-Series Summarisation Systems	Lecturer Function	Student Function	MO Function
Lecturer-adapted system	**243.82** (70.35)	51.99 (89.87)	114.12 (49.58)
Student-adapted system	72.54 (106.97)	**213.75** (59.45)	127.76 (52.09)
MO system	123.67 (72.66)	153.79 (56.61)	**164.84** (83.89)

Table 3: Average rewards (and standard deviation) assigned to summaries produced by the 3 systems. Bold signifies higher reward.

Summarisation Systems	Lecturer's Rating	Student's Rating
Lecturer-adapted	1st (2.15)*	3rd (1.97)
Student-adapted	1st (2.01)*	1st* (2.22)
MO	2nd, 3rd (1.81)	3rd (1.79)

Table 4: Mode of the ratings for each user group (*Mann-Whitney U test, $p < 0.05$, when comparing each system to the MO system).

that a student studied in all cases (i.e. when the hours_studied increased, decreased, or remained stable). This, however, does not factor heavily into the student's reward function.

Secondly, lecturers find it useful to give some advice to students who faced personal issues during the semester, such as advising them to talk to their mentor. Students, on the other hand, like reading about personal_issues only when the number of issues they faced was increasing over the semester, perhaps as this is the only trend that may affect their performance. Students seem to mostly prefer a feedback summary that mentions the understandability of the material when it increases which is positive feedback. Finally, the only factor that both groups agree on is that health_issues is negatively weighted and therefore not mentioned.

The MO reward function attempts to balance the preferences of the two user groups. Therefore, for this function, the coefficient for mentioning health_issues is also negative, however the other coefficients are smoothed providing neither strong negative or positive coefficients. This means that there is less variability (see Table 3) but that perhaps this function meets neither group's criteria.

5 Conclusion and Future Work

In conclusion, we presented a framework for developing and evaluating various reward functions for time-series summarisation of feedback. This framework has been validated in that both simulation and subjective studies show that each group does indeed prefer feedback generated using a highly tuned reward function, with lecturers being slightly more open to variation. Further investigation is required as to whether it is indeed possible to find middle ground between these two groups. Choices for one group may be negatively rated by the other and it might not be possible to find middle ground but it is worth investigating further other methods of reward function derivation using stronger feature selection methods, such as Principal Component Analysis.

References

Carole Ames. 1992. Classrooms: Goals, structures, and student motivation. *Journal of Educational Psychology*, 84(3):p261–71.

Chankong and Haimes. 1983. Multiobjective decision making theory and methodology. In *New York: Elsevier Science Publishing*.

Scotty D. Craig, Arthur C. Graesser, Jeremiah Sullins, and Barry Gholson. 2004. Affect and learning: an exploratory look into the role of affect in learning with autotutor. In *Journal of Educational Media, 29:241-250*.

Albert Gatt, Francois Portet, Ehud Reiter, James Hunter, Saad Mahamood, Wendy Moncur, and Somayajulu Sripada. 2009. From data to text in the neonatal intensive care unit: Using NLG technology for decision support and information management. In *Journal of AI Communications, 22:153-186*.

Dimitra Gkatzia, Helen Hastie, Srinivasan Janarthanam, and Oliver Lemon. 2013. Generating student feedback from time-series data using Reinforcement Learning. In *14th European Workshop in Natural Language Generation*.

Srinivasan Janarthanam and Oliver Lemon. 2010. Adaptive referring expression generation in spoken dialogue systems: Evaluation with real users. In *11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*.

K Papineni, S Roukos, T. Ward, and W. J Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *40th Annual meeting of the Association for Computational Linguistics*.

Natalie K. Person, Roger J. Kreuz, Rolf A. Zwaan, and Arthur C. Graesser. 1995. Pragmatics and pedagogy: Conversational rules and politeness strategies may inhibit effective tutoring. In *Journal of Cognition and Instruction, 13(2):161-188*.

Ehud Reiter and Robert Dale. 2000. Building natural language generation systems. In *Cambridge University Press*.

Verena Rieser, Oliver Lemon, and Xingkun Liu. 2010. Optimising information presentation for spoken dialogue systems. In *48th Annual Meeting of the Association for Computational Linguistics*.

Richart Sutton and Andrew Barto. 1998. Reinforcement learning. In *MIT Press*.

Cynthia A. Thompson, Mehmet H. Goker, and Pat Langley. 2004. A personalised system for conversational recommendations. In *Journal of Artificial Intelligence Research 21, 333-428*.

Marilyn Walker, Diane Litman, Candace Kamm, and Alicia Abella. 1997. PARADISE: A framework for evaluating spoken dialogue agents. In *35th Annual meeting of the Association for Computational Linguistics*.

Ingrid Zukerman and Diane Litman. 2001. Natural language processing and user modeling: Synergies and limitations. In *User Modeling and User-Adapted Interaction, 11(1-2), 129-158*.

One Sense per Tweeter ... and Other Lexical Semantic Tales of Twitter

Spandana Gella, Paul Cook and **Timothy Baldwin**
Department of Computing and Information Systems
The University of Melbourne
sgella@student.unimelb.edu.au, paulcook@unimelb.edu.au, tb@ldwin.net

Abstract

In recent years, microblogs such as Twitter have emerged as a new communication channel. Twitter in particular has become the target of a myriad of content-based applications including trend analysis and event detection, but there has been little fundamental work on the analysis of word usage patterns in this text type. In this paper — inspired by the one-sense-per-discourse heuristic of Gale et al. (1992) — we investigate user-level sense distributions, and detect strong support for "one sense per tweeter". As part of this, we construct a novel sense-tagged lexical sample dataset based on Twitter and a web corpus.

1 Introduction

Social media applications such as Twitter enable users from all over the world to create and share web content spontaneously. The resulting user-generated content has been identified as having potential in a myriad of applications including real-time event detection (Petrović et al., 2010), trend analysis (Lau et al., 2012) and natural disaster response co-ordination (Earle et al., 2010). However, the dynamism and conversational nature of the text contained in social media can cause problems for traditional NLP approaches such as parsing (Baldwin et al., 2013), meaning that most content-based approaches use simple keyword search or a bag-of-words representation of the text. This paper is a first step towards full lexical semantic analysis of social media text, in investigating the sense distribution of a range of polysemous words in Twitter and a general-purpose web corpus.

The primary finding of this paper is that there are strong user-level lexical semantic priors in Twitter, equivalent in strength to document-level lexical semantic priors, popularly termed the "one sense per discourse" heuristic (Gale et al., 1992). This has potential implications for future applications over Twitter which attempt to move beyond a simple string-based meaning representation to explicit lexical semantic analysis.

2 Related Work

The traditional approach to the analysis of word-level lexical semantics is via word sense disambiguation (WSD), where usages of a given word are mapped onto discrete "senses" in a pre-existing sense inventory (Navigli, 2009). The most popular sense inventory used in WSD research has been WordNet (Fellbaum, 1998), although its fine-grained sense distinctions have proven to be difficult to make for human annotators and WSD systems alike. This has resulted in a move towards more coarse-grained sense inventories (Palmer et al., 2004; Hovy et al., 2006; Navigli et al., 2007), or alternatively away from pre-existing sense inventories altogether, towards joint word sense induction (WSI) and disambiguation (Navigli and Vannella, 2013; Jurgens and Klapaftis, 2013).

Two heuristics that have proven highly powerful in WSD and WSI research are: (1) first sense tagging, and (2) one sense per discourse. First sense tagging is based on the observation that sense distributions tend to be Zipfian, such that if the predominant or "first" sense can be identified, simply tagging all occurrences of a given word with this sense can achieve high WSD accuracy (McCarthy et al., 2007). Unsurprisingly, there are significant differences in sense distributions across domains (cf. *cloud* in the COMPUTING and METEOROLOGICAL domains), motivating the need for unsupervised first sense learning over domain-specific corpora (Koeling et al., 2005).

One sense per discourse is the observation that a given word will often occur with a single sense across multiple usages in a single document (Gale

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 215–220,
Gothenburg, Sweden, April 26-30 2014.

et al., 1992). Gale et al. established the heuristic on the basis of 9 ambiguous words using a coarse-grained sense inventory, finding that the probability of a given pair of usages of a word taken from a given document having the same sense was 94%. However, Krovetz (1998) found that for a fine-grained sense inventory, only 67% of words exhibited the single-sense-per-discourse property for all documents in a corpus.

A radically different view on WSD is word usage similarity, whereby two usages of a given word are rated on a continuous scale for similarity, in isolation of any sense inventory (Erk et al., 2009). Gella et al. (2013) constructed a word usage similarity dataset for Twitter messages, and developed a topic modelling approach to the task, building on the work of Lui et al. (2012). To the best of our knowledge, this has been the only attempt to carry out explicit word-level lexical semantic analysis of Twitter text.

3 Dataset Construction

In order to study sense distributions of words in Twitter, we need a sense inventory to annotate against, and also a set of Twitter messages to annotate. Further, as a point of comparison for the sense distributions in Twitter, we require a second corpus; here we use the ukWaC (Ferraresi et al., 2008), a corpus built from web documents.

For the sense inventory, we chose the Macmillan English Dictionary Online[1] (MACMILLAN, hereafter), on the basis of: (1) its coarse-grained general-purpose sense distinctions, and (2) its regular update cycle (i.e. it contains many recently-emerged senses). These criteria are important in terms of inter-annotator agreement (especially as we crowdsourced the sense annotation, as described below) and also sense coverage. The other obvious candidate sense inventory which potentially satisfied these criteria was ONTONOTES (Hovy et al., 2006), but a preliminary sense-tagging exercise indicated that MACMILLAN better captured Twitter-specific usages.

Rather than annotating all words, we opted for a lexical sample of 20 polysemous nouns, as listed in Table 1. Our target nouns were selected to span the high- to mid-frequency range in both Twitter and the web corpus, and have at least 3 MACMILLAN senses. The average sense ambiguity is 5.5.

[1] `http://www.macmillandictionary.com`

band	*bar*	*case*	*charge*	*deal*
degree	*field*	*form*	*function*	*issue*
job	*light*	*match*	*panel*	*paper*
position	*post*	*rule*	*sign*	*track*

Table 1: The 20 target nouns used in this research

3.1 Data Sampling

We sampled tweets from a crawl made using the Twitter Streaming API from January 3, 2012 to February 29, 2012. The web corpus was built from ukWaC (Ferraresi et al., 2008), which was based on a crawl of the `.uk` domain from 2007. In contrast to ukWaC, the tweets are not restricted to documents from any particular country.

For both corpora, we first selected only the English documents using `langid.py`, an off-the-shelf language identification tool (Lui and Baldwin, 2012). We next identified documents which contained nominal usages of the target words, based on the POS tags supplied with the corpus in the case of ukWaC, and the output of the CMU ARK Twitter POS tagger v2.0 (Owoputi et al., 2012) in the case of Twitter.

For Twitter, we are interested in not just the overall lexical distribution of each target noun, but also per-user lexical distributions. As such, we construct two Twitter-based datasets: (1) $\text{TWITTER}_{\text{RAND}}$, a random sample of 100 usages of each target noun; and (2) $\text{TWITTER}_{\text{USER}}$, 5 usages of each target noun from each member of a random sample of 20 Twitter users. Naively selecting users for $\text{TWITTER}_{\text{USER}}$ without filtering resulted in a preponderance of messages from accounts that were clearly bots, e.g. from commercial sites with a single post per item advertised for sale, with artificially-skewed sense distributions. In order to obtain a more natural set of messages from "real" people, we introduced a number of user-level filters, including removing users who posted the same message with different user mentions or hashtags, and users who used the target nouns more than 50 times over a 2-week period. From the remaining users, we randomly selected 20 users per target noun, resulting in 20 nouns $\times$ 20 users $\times$ 5 messages = 2000 messages.

For ukWaC, we similarly constructed two datasets: (1) $\text{UKWAC}_{\text{RAND}}$, a random sample of 100 usages of each target noun; and (2) $\text{UKWAC}_{\text{DOC}}$, 5 usages of each target noun from 20 documents which contained that noun in at least

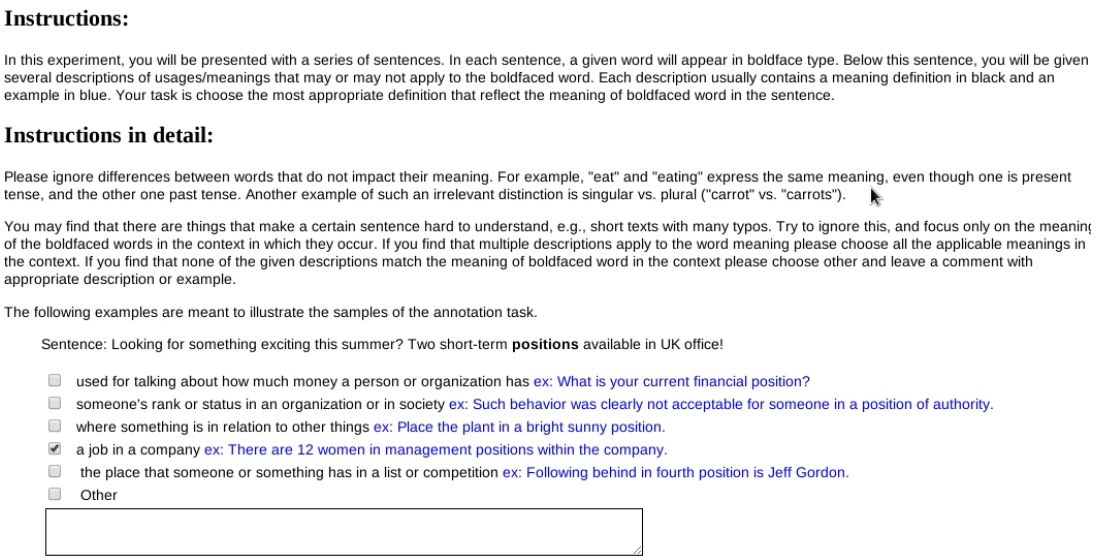


Figure 1: Screenshot of a sense annotation HIT for *position*

5 sentences. 5 such sentences were selected for annotation, resulting in a total of 20 nouns $\times$ 20 documents $\times$ 5 sentences = 2000 sentences.

3.2 Annotation Settings

We sense-tagged each of the four datasets using Amazon Mechanical Turk (AMT). Each Human Intelligence Task (HIT) comprised 5 occurrences of a given target noun, with the target noun highlighted in each. Sense definitions and an example sentence (where available) were provided from MACMILLAN. Turkers were free to select multiple sense labels where applicable, in line with best practice in sense labelling (Mihalcea et al., 2004). We also provided an "Other" sense option, in cases where none of the MACMILLAN senses were applicable to the current usage of the target noun. A screenshot of the annotation interface for a single usage is provided in Figure 1.

Of the five sentences in each HIT, one was a heldout example sentence for one of the senses of the target noun, taken from MACMILLAN. This gold-standard example was used exclusively for quality assurance purposes, and used to filter the annotations as follows:

1. Accept all HITs from Turkers whose gold-standard tagging accuracy was $\geq 80\%$;
2. Reject all HITs from Turkers whose gold-standard tagging accuracy was $\leq 20\%$;
3. Otherwise, accept single HITs with correct gold-standard sense tags, or at least 2/4 (non-gold-standard) annotations in common with Turkers who correctly annotated the gold-standard usage; reject any other HITs.

This style of quality assurance has been shown to be successful for sense tagging tasks on AMT (Bentivogli et al., 2011; Vuurens et al., 2011), and resulted in us accepting around 95% of HITs.

In total, the annotation was made up of 500 HITs (= 2000/4 usages per HIT) for each of the four datasets, each of which was annotated by 5 Turkers. Our analysis of sense distribution is based on only those HITs which were accepted in accordance with the above methodology, excluding the gold-standard items. We arrive at a single sense label per usage by unweighted voting across the annotations, allowing multiple votes from a single Turker in the case of multiple sense annotations. In this, the "Other" sense label is considered as a discrete sense label.

Relative to the majority sense, inter-annotator agreement post-filtering was respectably high in terms of Fleiss' kappa at $\kappa = 0.64$ for both UKWAC$_{\text{RAND}}$ and UKWAC$_{\text{DOC}}$. For TWITTER$_{\text{USER}}$, the agreement was actually higher at $\kappa = 0.71$, but for TWITTER$_{\text{RAND}}$ it was much weaker, $\kappa = 0.47$.

All four datasets have been released for public use: `http://www.csse.unimelb.edu.au/~tim/etc/twitter_sense.tgz`.

4 Analysis

In TWITTER$_{\text{USER}}$, the proportion of users who used a target noun with one sense across all 5 usages ranged from 7/20 for *form* to 20/20 for *degree*, at an average of 65%. That is, for 65% of users, a given noun (with average polysemy = 5.5 senses) is used with the same sense across 5 separate messages. For UKWAC$_{\text{DOC}}$ the proportion of documents with a single sense of a given target noun

	Partition	Agreement (%)
Gale et al. (1992)	document	94.4
TWITTER$_{\text{USER}}$	user	95.4
TWITTER$_{\text{USER}}$	—	62.9
TWITTER$_{\text{RAND}}$	—	55.1
UKWAC$_{\text{DOC}}$	document	94.2
UKWAC$_{\text{DOC}}$	—	65.9
UKWAC$_{\text{RAND}}$	—	60.2

Table 2: Pairwise agreement for each dataset, based on different partitions of the data ("—" indicates no partitioning, and exhaustive comparison)

across all usages ranged from 1/20 for *case* to 20/20 for *band*, at an average of 63%. As such, the one sense per tweeter heuristic is at least as strong as the one sense per discourse heuristic in UKWAC$_{\text{DOC}}$.

Looking back to the original work of Gale et al. (1992), it is important to realise that their reported agreement of 94% was calculated *pairwise* between usages in a given document. When we recalculate the agreement in TWITTER$_{\text{USER}}$ and UKWAC$_{\text{DOC}}$ using this methodology, as detailed in Table 2 (calculating pairwise agreement within partitions of the data based on "user" and "document", respectively), we see that the numbers for our datasets are very close to those of Gale et al. on the basis of more than twice as many nouns, and many more instances per noun. Moreover, the one sense per tweeter trend again appears to be slightly stronger than the one sense per discourse heuristic in UKWAC$_{\text{DOC}}$.

One possible interpretation of these results is that they are due to a single predominant sense, common to all users/documents rather than user-specific predominant senses. To test this hypothesis, we calculate the pairwise agreement for TWITTER$_{\text{USER}}$ and UKWAC$_{\text{DOC}}$ across all annotations (without partitioning on user/document), and also for TWITTER$_{\text{RAND}}$ and UKWAC$_{\text{RAND}}$. The results are, once again, presented in Table 2 (with partition indicated as "—" for the respective datasets), and are substantially lower in all cases ($< 66\%$). This indicates that the first sense preference varies considerably between users/documents. Note that the agreement is slightly lower for TWITTER$_{\text{RAND}}$ and UKWAC$_{\text{RAND}}$ simply because of the absence of the biasing effect for users/documents.

Comparing TWITTER$_{\text{RAND}}$ and UKWAC$_{\text{RAND}}$, there were marked differences in first sense preferences, with 8/20 of the target nouns having a different first sense across the two corpora. One surprising observation was that the sense distributions in UKWAC$_{\text{RAND}}$ were in general more skewed than in TWITTER$_{\text{RAND}}$, with the entropy of the sense distribution being lower (= more biased) in UKWAC$_{\text{RAND}}$ for 15/20 of the target nouns.

All datasets included instances of "Other" senses (i.e. usages which didn't conform to any of the MACMILLAN senses), with the highest relative such occurrence being in TWITTER$_{\text{RAND}}$ at 12.3%, as compared to 6.6% for UKWAC$_{\text{RAND}}$. Interestingly, the number of such usages in the user/document-biased datasets was around half these numbers, at 7.4% and 3.6% for TWITTER$_{\text{USER}}$ and UKWAC$_{\text{DOC}}$, respectively.

5 Discussion

It is worthwhile speculating why Twitter users would have such a strong tendency to use a given word with only one sense. This could arise in part due to patterns of user behaviour, in a given Twitter account being used predominantly to comment on a favourite sports team or political events, and as such is domain-driven. Alternatively, it can perhaps be explained by the "reactive" nature of Twitter, in that posts are often emotive responses to happenings in a user's life, and while different things excite different individuals, a given individual will tend to be excited by events of similar kinds. Clearly more research is required to test these hypotheses.

One highly promising direction for this research would be to overlay analysis of sense distributions with analysis of user profiles (e.g. Bergsma et al. (2013)), and test the impact of geospatial and sociolinguistic factors on sense preferences. We would also like to consider the impact of time on the one sense per tweeter heuristic, and consider whether "one sense per Twitter conversation" also holds.

To summarise, we have investigated sense distributions in Twitter and a general web corpus, over both a random sample of usages and a sample of usages from a single user/document. We found strong evidence for Twitter users to use a given word with a single sense, and also that individual first sense preferences differ between users, suggesting that methods for determining first senses on a per user basis could be valuable for lexical semantic analysis of tweets. Furthermore, we found that sense distributions in Twitter are overall less skewed than in a web corpus.

References

Timothy Baldwin, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang. 2013. How noisy social media text, how diffrnt social media sources? In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP 2013)*, pages 356–364, Nagoya, Japan.

Luisa Bentivogli, Marcello Federico, Giovanni Moretti, and Michael Paul. 2011. Getting expert quality from the crowd for machine translation evaluation. *Proceedings of the MT Summmit*, 13:521–528.

Shane Bergsma, Mark Dredze, Benjamin Van Durme, Theresa Wilson, and David Yarowsky. 2013. Broadly improving user classification via communication-based name and location clustering on Twitter. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, pages 1010–1019, Atlanta, USA.

Paul Earle, Michelle Guy, Richard Buckmaster, Chris Ostrum, Scott Horvath, and Amy Vaughan. 2010. OMG earthquake! can Twitter improve earthquake response? *Seismological Research Letters*, 81(2):246–251.

Katrin Erk, Diana McCarthy, and Nicholas Gaylord. 2009. Investigations on word senses and word usages. In *Proceedings of the Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing (ACL-IJCNLP 2009)*, pages 10–18, Singapore.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, USA.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop: Can we beat Google*, pages 47–54, Marrakech, Morocco.

William A Gale, Kenneth W Church, and David Yarowsky. 1992. One sense per discourse. In *Proceedings of the workshop on Speech and Natural Language*, pages 233–237.

Spandana Gella, Paul Cook, and Bo Han. 2013. Unsupervised word usage similarity in social media texts. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (*SEM 2013)*, pages 248–253, Atlanta, USA.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% solution. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 57–60, New York City, USA.

David Jurgens and Ioannis Klapaftis. 2013. Semeval-2013 task 13: Word sense induction for graded and non-graded senses. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 290–299, Atlanta, USA.

Rob Koeling, Diana McCarthy, and John Carroll. 2005. Domain-specific sense distributions and predominant sense acquisition. In *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing (EMNLP 2005)*, pages 419–426, Vancouver, Canada.

Robert Krovetz. 1998. More than one sense per discourse. *NEC Princeton NJ Labs., Research Memorandum*.

Jey Han Lau, Nigel Collier, and Timothy Baldwin. 2012. On-line trend analysis with topic models: #twitter trends detection topic model online. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 1519–1534, Mumbai, India.

Marco Lui and Timothy Baldwin. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012) Demo Session*, pages 25–30, Jeju, Republic of Korea.

Marco Lui, Timothy Baldwin, and Diana McCarthy. 2012. Unsupervised estimation of word usage similarity. In *Proceedings of the Australasian Language Technology Workshop 2012 (ALTW 2012)*, pages 33–41, Dunedin, New Zealand.

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2007. Unsupervised acquisition of predominant word senses. *Computational Linguistics*, 4(33):553–590.

Rada Mihalcea, Timothy Chklovski, and Adam Kilgarriff. 2004. The Senseval-3 English lexical sample task. In *Proceedings of Senseval-3: Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 25–28, Barcelona, Spain.

Roberto Navigli and Daniele Vannella. 2013. SemEval-2013 task 11: Word sense induction and disambiguation within an end-user application. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 193–201, Atlanta, USA.

Roberto Navigli, Kenneth C. Litkowski, and Orin Hargraves. 2007. SemEval-2007 task 07: Coarse-grained English all-words task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*, pages 30–35, Prague, Czech Republic.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys (CSUR)*, 41(2):10.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, and Nathan Schneider. 2012. Part-of-speech tagging for Twitter: Word clusters and other advances. Technical Report CMU-ML-12-107, Machine Learning Department, Carnegie Mellon University.

Martha Palmer, Olga Babko-Malaya, and Hoa Trang Dang. 2004. Different sense granularities for different applications. In *Proceedings of the HLT-NAACL 2004 Workshop: 2nd Workshop on Scalable Natural Language Understanding*, pages 49–56, Boston, USA.

Sasa Petrović, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to twitter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, pages 181–189, Los Angeles, USA.

Jeroen Vuurens, Arjen P de Vries, and Carsten Eickhoff. 2011. How much spam can you take? an analysis of crowdsourcing results to increase accuracy. In *Proc. ACM SIGIR Workshop on Crowdsourcing for Information Retrieval (CIR 2011)*, pages 21–26.

Zero subject detection for Polish

Mateusz Kopeć
Institute of Computer Science, Polish Academy of Sciences,
Jana Kazimierza 5, 01-248 Warsaw, Poland
m.kopec@ipipan.waw.pl

Abstract

This article reports on the first machine learning experiments on detection of null subjects in Polish. It emphasizes the role of zero subject detection as the part of mention detection – the initial step of end-to-end coreference resolution. Anaphora resolution is not studied in this article.

1 Introduction

Zero subject detection is an important issue for anaphora and coreference resolution for the null-subject languages, including all Balto-Slavic languages and most Romance languages. Their distinctive feature is the possibility for an independent clause to lack an explicit subject. Person, number, and/or gender agreement with the referent is indicated by the morphology of the verb:

(1) *Maria wróciła już z Francji. ØSpędziła tam miesiąc.*
"Maria came back from France. $\text{Ø}Had_{singular:feminine}$ spent a month there."

The recently created Polish Coreference Corpus[1] (PCC) (Ogrodniczuk et al., 2013) contains zero subject annotation. A markable representing the null subject is the verbal form following the position where the argument would have been expected. As tested on the development part of the corpus (described in detail later), omitting a personal pronoun is a frequent issue in the Polish language – about 30% of verbs do not have explicit subjects. Russo et al. (2012) reports similar figures for Italian (30.42%) and Spanish (41.17%).

Moreover, these null subjects are often part of large coreference clusters – the average size of a non-singleton coreference cluster in the development subcorpus was 3.56 mentions. At the same time, the non-singleton coreference cluster containing at least one zero subject had on average 5.89 mentions.

A mention detection module heavily influences the final coreference resolution score of an end-to-end coreference resolution system. In Ogrodniczuk and Kopeć (2011a) the system working on gold mentions achieved 82.90% F1 BLANC (Recasens and Hovy, 2011), whereas on system mentions the result dropped to 38.13% (the zero subject detection module was not implemented).

The aim of this paper is to find a method of automatic zero subject detection to improve the accuracy of mention detection as the initial step of coreference resolution.

2 Related Work

We present some of the most recent articles about machine learning zero subject detection.

Rello et al. (2012b) describes a Brazilian Portuguese corpus with 5665 finite verbs total, out of which 77% have an explicit subject, 21% a zero pronoun and 2% are impersonal constructions. They extract various verb, clause and neighboring token features for each verb occurrence and classify it into one of these 3 classes, achieving 83.04% accuracy of a decision tree learning classifier, better than the baseline result of the *Palavras* parser. A very similar study is conducted also for Spanish (Rello et al., 2012a), with the best result of the lazy learning classifier K^* (Cleary and Trigg, 1995) of 87.6% accuracy, outperforming the baseline of *Connexor* parser.

Chinese zero pronoun detection and resolution is presented by Zhao and Ng (2007). Features for zero pronoun identification consider mainly the gold standard parse tree structure. Their training corpus contained only 343 zero pronouns, as compared to 10098 verbs with explicit subjects – for Chinese, the phenomenon is much less frequent than for Polish or Spanish. Therefore they weigh

[1] Publicly available at http://zil.ipipan.waw.pl/PolishCoreferenceCorpus.

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 221–225,
Gothenburg, Sweden, April 26-30 2014.

positive and negative examples to get the balance between precision and recall – the best result of 50.9% F_1 measure for positive to negative example weight ratio of 8:1 is reported.

A study for the Romanian language (Mihaila et al., 2011) describes a corpus consisting of 2741 sentences and 997 zero pronouns. Class imbalance is solved by training machine learning algorithms on all positive examples (zero pronouns) and the same number of negative examples (sampled from the corpus). Features used consider morphosyntactic information about the verb, precedence of the reflective pronoun "se" and the number of verbs in the sentence. Their best ensemble classifier scored 74.5% accuracy.

Only a few studies (for example (Broda et al., 2012; Ogrodniczuk and Kopeć, 2011b; Kopeć and Ogrodniczuk, 2012)) consider the problem of rule-based or machine learning coreference resolution for the Polish language, however these attempts leave zero subject detection as a non-trivial task for further study.

3 Problem statement

Table 1 presents part of speech definitions assumed in this article, based on the book about the National Corpus of Polish (Przepiórkowski et al., 2012). Coarse-grained POS indicates whether a word with a given part of speech may be a subject (*Noun*) or a verb (*Verb*) in a sentence. The last four columns present which morphosyntactic information is available for each part of speech. There are few differences in this definition with respect to the original approach in the book:

- We treat numerals, gerunds and pronouns as *Nouns* – because they are frequently subjects of the sentence and have the same morphosyntactic information as "standard" nouns.
- We do not consider *siebie* ("self", traditionally treated as pronoun) as a *Noun*, as it cannot be a subject.
- Tags: *impt, imps, inf, pcon, pant, pact, ppas, pred*, which are traditionally considered verb tags, are not treated by us as *Verbs*, because they cannot have a subject.

With such a definition of parts of speech, our task may be stated as follows: given a clause with a *Verb*, decide whether the clause contains a *Noun*

Coarse-grained POS	POS	Tag	Number	Case	Gender	Person
Noun	Noun	subst	+	+	+	
	Depreciative form	depr	+	+	+	
	Main numeral	num	+	+	+	
	Collective numeral	numcol	+	+	+	
	Gerund	ger	+	+	+	
	Personal pronoun – 1st, 2nd person	ppron12	+	+	+	+
	Personal pronoun – 3rd person	ppron3	+	+	+	+
Verb	Non-past form	fin	+			+
	Future *być*	bedzie	+			+
	Agglutinate *być*	aglt	+			+
	L-participle	praet	+		+	
	winien-like verb	winien	+		+	

Table 1: Parts of speech

which is the *Verb's* explicit subject. From now on in this paper, the words "noun" and "verb" have the meaning of *Noun* and *Verb*, respectively. In this study, we do not try to handle the cases of subjects not being nouns, as judging from our observations, it is very infrequent. We do take into account in our solution the cases of the subject not in the *nominative* case, as in the example:

(2) *Pieniędzy*$_{noun:genitive}$ *nie starczy dla wszystkich.*

"There wouldn't be enough money for everyone."

It is worth noting that Polish is a free-word-order language, therefore there are many possible places for the subject to appear, with respect to the position of the verb.

As the corpus has only automatic morphosyntactic information available (provided by the PANTERA tagger (Acedański, 2010)), not corrected by the coreference annotators, the only verbs considered in this study are the ones found by the tagger. If such a verb was marked as a mention by the coreference annotator (*verb mention* in table 2), it is a positive example for our machine learning study, otherwise a negative one. Sentence and clause segmentation in the corpus was also automatic. We are aware that the corpus used for the study was not perfectly suited for the task – verbs with a zero subject are not marked there explicitly, but can only be found based on automatic tagging. However the tagging error of detecting verbs is reported as not higher than 0.04% (for the *fin* tag, see (Acedański, 2010) for details), so we consider the resource sufficiently correct.

4 Development and evaluation data

Each text of the Polish Coreference Corpus is a 250-350 word sample, consisting of full, subsequent paragraphs extracted from a larger text. Text genres balance correspond to the National Corpus

Corpus	# texts	# sentences	# tokens	# verbs	# mentions	# verb mentions
Development	390	6481	110379	10801	37250	3104
Evaluation	389	6737	110474	11000	37167	3106
Total	779	13218	220853	21801	74417	6210

Table 2: Zero subject study data statistics

of Polish (Przepiórkowski et al., 2012). At the time this study started, 779 out of 1773 texts (randomly chosen) of the Polish Coreference Corpus were already manually annotated. Annotated texts were randomly split into two equal-sized subcorpora for development and evaluation. Their detailed statistics are presented in Table 2.

4.1 Inter-annotator agreement

210 texts of the Polish Coreference Corpus were annotated independently by two annotators. This part was analyzed for the inter-annotator agreement of deciding if a verb has a zero subject or not. In the data there were 5879 verbs total, for which observed agreement yielded 92.57%. Agreement expected by chance (assuming a per annotator chance annotation probability distribution) equalled 57.52%, therefore chance-corrected Cohen's κ for the task equalled 82.51%.

4.2 Results of full dependency parsing

The first Polish dependency parser was recently developed and described by Wróblewska (2012). The author reports 71% LAS[2] and 75.2% UAS[3] performance of this parser. This parser was used to detect null subjects – every verb lacking the dependency relation of the subject type (`subj`) was marked as missing the subject. This baseline method achieved accuracy of 67.23%, precision of 46.53%, recall of 90.47% and F_1 equal to 61.45%. These results are worse than a simple majority baseline classifier, therefore current state-of-the-art Polish dependency parsing is not a satisfactory solution to the task stated in this article.

5 Features

Based on a number of experiments on the development corpus, we chose a number of features presented in table 3.

Subject candidate existence features from the bottom of the table 3 use variables: c_1, c_2 and w. Separate feature was generated for each combination of these three variables. The variable w represents the window around the verb, with following values: the clause containing the verb, the sentence containing the verb, windows of 1 to 5 tokens before the verb, windows of 1 to 5 tokens after the verb, windows of 1 to 5 tokens both before and after the verb. Variable c_1 represents compatibility of noun and verb, with values being any nonempty subset of the set of following conditions: case of the noun equal to *nominative* (NOM), number agreement with the verb (NUM), person or gender agreement (POG), depending on which was available to check, see Table 1. Variable c_2 is similar to c_1, with the following values: {NOM}, {POG}, {NOM, POG}.

Feature	Type
Verb features	
number of the verb – to help with cases of plural verbs having two or more singular nouns as subject	nominal
tag of the verb – as it may happen, that some parts of speech behave differently	boolean
is the verb on the pseudo-verbs list extracted from (Świdziński, 1994) – i.e. may not require a subject	boolean
Neighboring token features	
tag of the next token	nominal
tag of the previous token	nominal
is the previous tag equal to *praet* – a redundant feature to the previous one, but it should help with the cases like: $\ldots$ *była*$_{praet}$*m*$_{aglt:pri}$ $\ldots$ "$\ldots$ *(I) was* $\ldots$" when we split a word into a L-participle and agglutinate. Annotation guidelines were to only mark the agglutinate as a mention, when the verb does not have an explicit subject	boolean
does one of the previous two tokens have the *pred* tag – should allow detecting examples similar to: *Można*$_{pred}$ *się było*$_{praet}$ *tego spodziewać.* "$\ldots$ *It could have been expected.* $\ldots$" *Trzeba*$_{pred}$ *było*$_{praet}$ *myśleć wcześniej.* "*(One) should have thought before.*" when *było ("have")* cannot have subject, as it is part of an impersonal construction	boolean
is the next tag *inf* – similar role to the previous feature, as in: *Wtedy należy*$_{fin}$ *poprosić*$_{inf}$. "*(One) should then ask for it.*" when *należy ("one should")* cannot have a subject	boolean
is the previous token a comma	boolean
Length features	
number of tokens in the sentence (following the hypothesis, that the shorter the sentence/clause, the less likely for the subject to appear)	numerical
number of tokens in the clause with the verb	numerical
Subject candidate existence features	
existence of a noun not preceded by *jak/jako ("as")* in window w fulfilling conditions from set c_1	boolean
existence of at least two nouns not preceded by *jak/jako ("as")* in window w both fulfilling conditions from set c_2	boolean

Table 3: Features

6 Evaluation

Presented features were used to train a machine learning algorithm. We chose the JRip implementation of RIPPER (Cohen, 1995) from WEKA (Hall et al., 2009) for the possibility to interpret the rules, which is outside of the scope of this paper.

6.1 Accuracy on the development corpus

A baseline model which always predicts that a verb has an explicit subject achieves 71.13% ac-

[2]Labeled attachment score – the percentage of tokens that are assigned a correct head and a correct dependency type.

[3]Unlabeled attachment score – the percentage of tokens that are assigned a correct head.

		True values	
		null subject	explicit subject
Predictions	null subject	2093	815
	explicit subject	1013	7079

Table 4: Confusion matrix

curacy on the development data. The upper bound of the ITA (as stated earlier) is around 92.57% accuracy.

We used 10-fold cross-validation which was repeated 10 times with different random seeds for training and train/test splits. The average from the total of 100 trials (each cross-validation split separately) was equal to 82.74%, with standard deviation of 1.27%. As the Shapiro-Wilk (1965) test for normality for this data gives p-value of 0.38, it may be assumed that it follows the normal distribution. In that case, the 95% confidence interval for the accuracy is equal to [82.49%, 82.99%].

6.2 Accuracy on the evaluation corpus

The evaluation corpus was used only for two experiments presented below: to calculate accuracy and learning curve of the developed solution.

We used the model learnt on the development corpus and tested it on the evaluation corpus, achieving 83.38% accuracy. A majority classifier would achieve 71.76% accuracy on this corpus. The confusion matrix is depicted in Table 4. For finding the null subjects, recall of 67.39% and precision of 71.97% gives F_1 measure of 69.60%.

6.3 Learning curve

To test how the number of training examples influences the quality of the trained classifier, we used subsets of the development corpus of various sizes as training sets. The test set was the same in all cases (the evaluation corpus). Proportions of the examples used ranged from 5% to 100% of the development corpus, each proportion was tested 10 times to provide an estimation of variance. For example, to evaluate the efficiency of the classifier trained on 5% of the training examples, we randomly sampled 5% of the examples, trained the classifier and tested it on the full evaluation corpus. Then we repeated it another 9 times, randomly choosing a different 5% portion of the examples for training.

Again the Shapiro-Wilk test was taken to assess the normality of results for each proportion, out of 19 proportions tested (the proportion of 1 was of course not tested for normality), only 3 had p-value less than 0.1, therefore we assumed that the data is distributed approximately normally. The 95% confidence intervals of the classifiers trained on a given proportion of the development corpus are shown in the Figure 1. The algorithm clearly benefits from having more training examples. We observe that the curve is generally of the desired shape, yet it flattens when approaching the full training set used. It may suggest that the developed solution would not be able to significantly exceed 84%, even given more training examples.

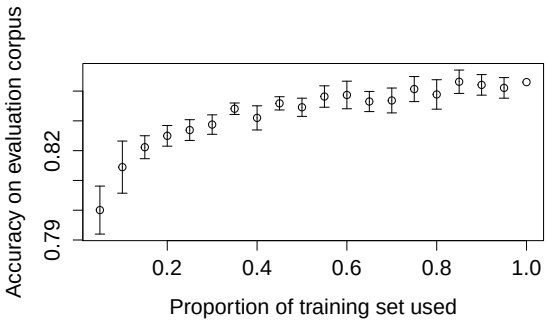


Figure 1: Learning curve

7 Conclusions and future work

This article presented an efficient zero subject detection module for Polish. We highlighted some difficult examples to take into account and proposed a solution for the Polish language.

The achieved accuracy of 83.38% significantly exceeds the baseline of majority tagging, equal to 71.76%, but there is still room for improvement, as the upper bound of 92.57% was computed. The achieved result for the task of null subject detection looks promising for the application in mention detection for coreference resolution.

The invented solution needs to be incorporated in a complete coreference resolver for Polish and evaluated for the extent to which using such an advanced separate classifier for zero subject detection improves the mention detection and, furthermore, end-to-end coreference resolution accuracy.

Acknowledgements

The work reported here was cofounded by the Computer-based methods for coreference resolution in Polish texts project financed by the Polish National Science Centre (contract number 6505/B/T02/2011/40) and by the European Union from resources of the European Social Fund. Project PO KL „Information technologies: Research and their interdisciplinary applications”.

References

Szymon Acedański. 2010. A Morphosyntactic Brill Tagger for Inflectional Languages. In Hrafn Loftsson, Eiríkur Rögnvaldsson, and Sigrún Helgadóttir, editors, *Advances in Natural Language Processing*, volume 6233 of *Lecture Notes in Computer Science*, pages 3–14. Springer.

Bartosz Broda, Łukasz Burdka, and Marek Maziarz. 2012. IKAR: An Improved Kit for Anaphora Resolution for Polish. In *COLING (Demos)*, pages 25–32.

John G. Cleary and Leonard E. Trigg. 1995. K*: An instance-based learner using an entropic distance measure. In *In Proceedings of the 12th International Conference on Machine Learning*, pages 108–114. Morgan Kaufmann.

William W. Cohen. 1995. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123. Morgan Kaufmann.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The WEKA data mining software: an update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November.

Mateusz Kopeć and Maciej Ogrodniczuk. 2012. Creating a Coreference Resolution System for Polish. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012*, pages 192–195, Istanbul, Turkey. ELRA.

Claudiu Mihaila, Iustina Ilisei, and Diana Inkpen. 2011. Zero Pronominal Anaphora Resolution for the Romanian Language. *Research Journal on Computer Science and Computer Engineering with Applications" POLIBITS*, 42.

Maciej Ogrodniczuk and Mateusz Kopeć. 2011a. End-to-end coreference resolution baseline system for Polish. In Zygmunt Vetulani, editor, *Proceedings of the 5th Language & Technology Conference: Human Language Technologies as a Challenge for Computer Science and Linguistics*, pages 167–171, Poznań, Poland.

Maciej Ogrodniczuk and Mateusz Kopeć. 2011b. Rule-based coreference resolution module for Polish. In *Proceedings of the 8th Discourse Anaphora and Anaphor Resolution Colloquium (DAARC 2011)*, pages 191–200, Faro, Portugal.

Maciej Ogrodniczuk, Katarzyna Głowińska, Mateusz Kopeć, Agata Savary, and Magdalena Zawisławska. 2013. Polish coreference corpus. pages 494–498.

Adam Przepiórkowski, Mirosław Bańko, Rafał L. Górski, and Barbara Lewandowska-Tomaszczyk, editors. 2012. *Narodowy Korpus Języka Polskiego [Eng.: National Corpus of Polish]*. Wydawnictwo Naukowe PWN, Warsaw.

Marta Recasens and E. Hovy. 2011. BLANC: Implementing the Rand index for coreference evaluation. pages 485–510.

Luz Rello, Ricardo Baeza-Yates, and Ruslan Mitkov. 2012a. Elliphant: Improved Automatic Detection of Zero Subjects and Impersonal Constructions in Spanish. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 706–715, Avignon, France, April. Association for Computational Linguistics.

Luz Rello, Gabriela Ferraro, and Iria Gayo. 2012b. A First Approach to the Automatic Detection of Zero Subjects and Impersonal Constructions in Portuguese. *Procesamiento del Lenguaje Natural*, 49:163–170.

Lorenza Russo, Sharid Loáiciga, and Asheesh Gulati. 2012. Improving machine translation of null subjects in Italian and Spanish. In *Proceedings of the Student Research Workshop at the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 81–89, Avignon, France, April. Association for Computational Linguistics.

S. S. Shapiro and M. B. Wilk. 1965. An analysis of variance test for normality (complete samples). *Biometrika*, 52(3/4):591–611, Dec.

Marek Świdziński. 1994. Syntactic dictionary of polish verbs.

Alina Wróblewska. 2012. Polish dependency bank. *Linguistic Issues in Language Technology*, 7(1).

Shanheng Zhao and Hwee Tou Ng. 2007. Identification and Resolution of Chinese Zero Pronouns: A Machine Learning Approach. In *EMNLP-CoNLL*, pages 541–550. ACL.

Crowdsourcing Annotation of Non-Local Semantic Roles

Parvin Sadat Feizabadi
Institut für Computerlinguistik
Heidelberg University
69120 Heidelberg, Germany
feizabadi@cl.uni-heidelberg.de

Sebastian Padó
Institut für Maschinelle Sprachverarbeitung
Stuttgart University
70569 Stuttgart, Germany
pado@ims.uni-stuttgart.de

Abstract

This paper reports on a study of crowdsourcing the annotation of *non-local* (or *implicit*) frame-semantic roles, i.e., roles that are realized in the previous discourse context. We describe two annotation setups (marking and gap filling) and find that gap filling works considerably better, attaining an acceptable quality relatively cheaply. The produced data is available for research purposes.

1 Introduction

In the last years, crowdsourcing, e.g., using Amazon's Mechanical Turk platform, has been used to collect data for a range of NLP tasks, e.g., MT evaluation (Callison-Burch, 2009), sentiment analysis (Mellebeek et al., 2010), and student answer rating (Heilman and Smith, 2010). Frame-semantic role annotation (FSRA) is a task that requires more linguistic expertise than most data collection tasks realized with crowdsourcing; nevertheless it is also a crucial prerequisite for high-performance frame-semantic role labeling (SRL) systems (Das et al., 2014). Thus, there are some studies that have investigated FSRA as a crowdsourcing task. It can be separated into two parts: First, choosing the frame evoked by a given predicate in a sentence; second, assigning the semantic roles associated with the chosen frame. Hong and Baker (2011) have recently addressed the first step, experimenting with various ways of presenting the task. Fossati et al. (2013) have considered both steps and operationalized them separately and jointly, finding the best results when a single annotation task is presented to turkers (due to the interdependence of the two steps) and when the semantic role description are simplified. Both studies conclude that crowdsourcing can produce usable results for FSRA but requires careful design. Our study extends these previous studies to the phenomenon of implicit (non-locally realized) semantic roles where annotators are presented with a target sentence in paragraph context, and have to decide for every role whether it is realized in the target sentence, elsewhere in the paragraph, or not at all. Our results shows that implicit roles can be annotated as well as locally realized roles in a crowdsourcing setup, again provided that good design choices are taken.

2 Implicit Semantic Roles

Implicit or non-locally realized semantic roles occur when arguments of a predicate are understood although not expressed in its direct syntactic neighborhood. FrameNet (Fillmore et al., 2003) distinguishes between indefinite non-instantiations (INIs), which are interpreted generically; definite non-instantiations (DNIs), which can often be identified with expressions from the previous context; and constructional non-instantiations (CNI), e.g., passives. For instance, in the following example, the GOAL of the predicate "reached" is realized locally, the SOURCE is a non-locally realized DNI, and the PATH is an INI and not realized at all.

(1) Phileas Fogg, having shut the door of [SOURCE his house] at half-past eleven, and having put his right foot before his left five hundred and seventy-five times, and his left foot before his right five hundred and seventy-six times, **reached** [GOAL the Reform Club].

Implicit roles play an important role in discourse comprehension and coherence (Burchardt et al., 2005) and have found increasing attention over the

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 226–230,
Gothenburg, Sweden, April 26-30 2014.

last years. The development was kickstarted by the creation of a corpus of non-local frame-semantic roles for the SemEval 2010 Task 10 (Ruppenhofer et al., 2010), which still serves as a de facto standard. A number of systems perform SRL for non-local roles (Chen et al., 2010; Silberer and Frank, 2012; Laparra and Rigau, 2013), but the obtained results are still far from satisfactory, with the best reported F-Score at 0.19. The main reason is data sparsity: Due to the small size of the dataset (just 438 sentences), every predicate occurs only a small number of times. Crowdsourcing can be an attractive strategy to acquire more annotations.

3 Experimental Setup

3.1 Domain

Our emphasis is on evaluating the annotation of implicit roles. We reduce complexity by limiting the number of frames and roles like earlier studies (Hong and Baker, 2011; Fossati et al., 2013). We focus on verbs from the MOTION and POSITION frames, which realize a common set of location roles (PLACE OF EVENT, SOURCE, GOAL, PATH). This makes the task more uniform and allows us to skip frame annotation. Information about spatial relations, provided by such verbs, can be useful for many NLP tasks which reason about spatial information, e.g. systems generating textual descriptions from visual data, robot navigation tasks, and geographical information systems or GIS (Kordjamshidi et al., 2012).

3.2 Corpus

We chose the novel "Around the World in Eighty Days" by Jules Verne, annotating the ten most frequent predicates meeting the conditions described above for annotation (*reach, arrive, descend, rush, follow, approach, send, cross, escape, pass*). A post-hoc analysis later showed that each instance of these predicates has on average 0.67 implicit roles identifiable in previous context, which underlines the relevance of annotating such cases. Metaphorical uses were discarded before annotation, which left an average 38.4 instances for each predicate.

4 Annotation and Agreement

We decided to present target sentences with three sentences of previous context, as a compromise between reading overhead and coverage of non-local roles: For nominalizations, the three previous sentences cover over 85% of all non-local roles (Gerber and Chai, 2012). An example and the detailed description of the task were provided to the annotators through external links. We experimented with two alternatives: annotation as a *marking task* and as a *gap filling task* (explained below). Each HIT was annotated by five turkers who were asked to annotate both local and non-local roles, since identification of local roles is necessary for reliable tagging of non-local roles.

	Source	Goal	Path	Place
Exact Match	0.35	0.44	0.48	0.24
Overlap	0.35	0.46	0.52	0.27

Table 1: Raw agreement among annotators in the "marking" task

4.1 Marking Task

Our rationale was to make the task as comprehensible as possible for non-experts. In each HIT, the target predicate in its context was shown in boldface and the annotators were asked to answer four questions about "the event in bold": (a) where does the event take place?; (b) what is its starting point?; (c) what is its end point?; (d) which path is used? For every question, turkers were asked to either mark a text span (shown in a non-editable field below the question) or click a button labeled "not found in the text". The goals of this setup were (a) to minimize annotation effort, and (b) to make the task as layman-compatible as possible, following Fossati et al.'s (2013) observation that linguistic definitions can harm results.

After annotating some instances, we computed raw inter-annotator agreement (IAA). Table 1 shows IAA among turkers in two conditions (average pairwise Exact Match and word-based Overlap) overall annotations for the first 49 instances.[1] The overall IAA is 37.9% (Exact Match) and 40.1% (Overlap). We found these results to be too low to continue this approach. The low results for Overlap indicate that the problems cannot be due mainly to differences in the marked spans. Indeed, an analysis showed that the main reason was that annotators were often confused by the presence of multiple predicates in the paragraph. Consequently, many answers marked roles pertaining not to the bolded target predicate but to other predicates, such as (2).

(2) Leaving Bombay, it passes through Sal-

[1] Kappa is not applicable since we have a large number of disjoint annotators.

	Source	Goal	Path	Place
Exact Match	0.46	0.46	0.56	0.30
Overlap	0.50	0.54	0.58	0.38

Table 2: Raw agreement among annotators in the "gap filling" task

> cette, **crossing** to the continent opposite Tannah, goes over the chain of the Western Ghauts, [...] and, descending southeastward by Burdivan and the French town of Chandernagor, has its terminus at Calcutta.

Annotators would be expected to annotate *the continent opposite Tannah* as the goal of crossing, but some annotated *Calcutta*, the final destination of the chain of motion events described.

4.2 Gap Filling Task

Seeing that the marking task did not constrain the interpretation of the turkers sufficiently, we moved to a second setup, gap filling, with the aim of focussing the turkers' attention to a single predicate rather than the complete set of predicates present in the text shown. In this task, the annotators were asked to complete the sentence by filling in the blanks in two sentences:

1. [Agent] [Event+ed] from ... to ... through ... path.

2. The whole event took place in/at ...

The first sentence corresponds to annotations of the SOURCE, GOAL, and PATH roles; the second one of the PLACE role. The rationale is that the presence of the predicate in the sentence focuses the turkers' attention on the predicate's actual roles. Annotators could leave gaps empty (in the case of unrealized roles), and we asked them to remain as close to the original material as possible, that is, avoid paraphrases. Perfect copying is not always possible, due to grammatical constraints.

Table 2 shows the IAA for this design. We see that even though the gap filling introduced a new source of variability (namely, the need for annotators to copy text), the IAA improves considerably, by up to 11% in Exact Match and 15% in Overlap. The new overall IAAs are 44.7% (+6.8%) and 50.2% (+10.1%), respectively. Overall, the numbers are still fairly low. However, note that these IAA numbers among turkers are a lower bound for the agreement between a "canonical" version of the turkers' annotation (see Section 5) and an ideal gold standard. Additionally, a data analysis showed that in the gap filling setup, many of the disagreements are more well-behaved: unsurprisingly, they are often cases where annotators disagree on the exact range of the string to fill into the gap. Consider the following example:

(3) Skillful detectives have been **sent** to all the principal ports of America and the Continent, and he'll be a clever fellow if he slips through their fingers."

Arguably, experts would annotate *all the principal ports of America and the Continent* as the GOAL role of **sent**. Turkers however annotated different spans, including *all the principal ports of America*, *ports*, as well as the "correct" span. The lowest IAA is found for the place role. While it is possible that our setup which required turkers to consider a second sentence to annotate place contributes to the overall difficulty, our data analysis indicates that the main problem is the more vague nature of PLACE compared to the other roles which made it more difficult for annotators to tag consistently. Consider Example (1): the PLACE could be, among other things, *the City*, *London*, *England*, etc. The large number of locations in the novel is a compounding factor. We found that for some predicates (e.g. *arrive, reach*), many turkers attempted to resolve the ambiguity by (erroneously) annotating the same text as both GOAL and PLACE, which runs counter to the FrameNet guidelines.

5 Canonicalization

We still need to compute a "canonical" annotation that combines the five turker's annotations. First, we need to decide whether a role should be realized or left unrealized (i.e., INI, CNI, or DNI but not in the presented context). Second, we need to decide on a span for realized roles. Canonicalization in crowdsourcing often assumes a majority principle, accepting the analysis proposed by most turkers. We found it necessary to be more flexible. Regarding realization, a manual analysis of a few instances showed that cases of two turker annotations with non-empty overlap could be accepted as non-local roles. That is, turkers frequently miss non-local roles, but if two out of five annotate an overlapping span with the same role, this is reasonable evidence. Regarding the role's span, we used the consensus

	Source	Goal	Path	Place
Exact Match	0.72	0.67	0.82	0.50
Overlap	0.72	0.69	0.82	0.54

Table 3: Raw agreement between canonical crowdsourcing annotation and expert annotation by role

	Local	Non-Local	Unrealized
Exact Match	0.66	0.66	0.69
Overlap	0.69	0.70	0.69

Table 4: Raw agreement between canonical annotation and expert annotation by realization status

span if it existed, and the maximal (union) span otherwise, given that some turkers filled the gaps just with head words and not complete constituents. To test the quality of the canonical annotation, one of the authors had previously annotated 100 random instances that were also presented to the turkers. We consider the result to be an expert annotation approximating a gold standard and use it to judge the quality of the canonical turker annotations. The results are shown in Table 3.

The overall raw agreement numbers are 67.80% (Exact Match) and 69.34% (Overlap). As we had hoped, the agreement between the canonical crowdsourcing annotation and the expert annotation is again substantially higher than the IAA among turkers. Again, we see the highest numbers for path (the most specific role) and the lowest numbers for place (the least specific role).

To assess whether the number obtained in table 3 are sensitive to realization status (explicit, implicit or unrealized), we broke down the agreement numbers by realization status. Somewhat to our (positive) surprise, the results in Table 4 indicate that non-locally realized roles are annotated ablut as reliably as locally realized ones. Except for the ill-defined PLACE role, our reliability is comparable to Fossati et al. (2013). Given the more difficult nature of the task (annotators are given more context and have to make a more difficult decision), we consider this a promising result.

6 Final Dataset and Cost

The final dataset consists of 384 predicate instances.[2] With four roles per predicate, a total of 1536 roles could have been realized. We found that more than half (60%) of the roles remained unrealized even in context. 23% of the roles were realized locally, and 17% non-locally. The distribution over locally realized, non-locally realized, and unrealized roles varies considerably among the four roles that we consider. GOAL has the highest percentage of realized roles overall (unrealized only for 34% of all predicate instances), and at the same time the highest ratio of locally realized roles (48% locally realized, 18% non-locally). This corresponds well to FrameNet's predictions about our chosen predicates which realize the Goal role generally as the direct object (*reach*) or an obligatory prepositional phrase (*arrive*). In contrast, SOURCE is realized only for 36% of all instances, and then predominantly non-locally (24% non-local vs. 12% local). This shows once more that a substantial part of predicate-argument structure must be recovered from previous discourse context.

On average, each HIT page was annotated in 1 minute and 48 seconds, which means 27 seconds per each role and a total of 60 hours for the whole annotation. We paid 0.15 USD for each HIT. Since the number of roles in all HITs was fixed to four (source, goal, path and place), each role cost 0.04 USD, which corresponds to about USD 0.19 for every canonical role annotation. This is about twice the amount paid by Fossati et al. and reflects the increased effort inherent in a task that involves discourse context.

7 Conclusion

This paper presented a study on crowdsourcing the annotation of non-local semantic roles in discourse context, comparing a marking and a gap filling setup. We found that gap filling is the more reliable choice since the repetition of the predicate helps focusing the turkers' attention on the roles at hand rather than understanding of the global text. Thus, the semantic role-based crowdsourcing approach of Fossati et al. (2013) appears to be generalizable to the area of non-locally realized roles, provided that the task is defined suitably. Our results also support Fossati et al.'s observation that reliable annotations can be obtained without providing definitions of semantic roles. However, we also find large differences among semantic roles. Some (like PATH) can be annotated reliably and should be usable to train or improve SRL systems. Others (like PLACE) are defined so vaguely that it is unclear how usable their annotations are.

[2] It can be downloaded for research purposes from `http://www.cl.uni-heidelberg.de/~feizabadi/res.mhtml`

References

Aljoscha Burchardt, Anette Frank, and Manfred Pinkal. 2005. Building text meaning representations from contextually related frames – a case study. In *Proceedings of the International Workshop on Computational Semantics*, pages 66–77, Tilburg, Netherlands.

Chris Callison-Burch. 2009. Fast, cheap, and creative: evaluating translation quality using Amazon's Mechanical Turk. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 286–295, Singapore.

Desai Chen, Nathan Schneider, Dipanjan Das, and Noah A. Smith. 2010. Semafor: Frame argument resolution with log-linear models. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 264–267, Uppsala, Sweden.

Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. 2014. Frame-semantic parsing. *Computational Linguistics*. To appear.

Charles J Fillmore, Christopher R Johnson, and Miriam R L Petruck. 2003. Background to FrameNet. *International Journal of Lexicography*, 16(3):235–250.

Marco Fossati, Claudio Giuliano, Sara Tonelli, and Fondazione Bruno Kessler. 2013. Outsourcing FrameNet to the Crowd. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 742–747, Sofia, Bulgaria.

Matthew Gerber and Joyce Y Chai. 2012. Semantic role labeling of implicit arguments for nominal predicates. *Computational Linguistics*, 38(4):755–798.

Michael Heilman and Noah A Smith. 2010. Rating computer-generated questions with Mechanical Turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 35–40, Los Angeles, CA.

Jisup Hong and Collin F. Baker. 2011. How good is the crowd at "real" WSD? In *Proceedings of the 5th Linguistic Annotation Workshop*, pages 30–37, Portland, Oregon, USA.

Parisa Kordjamshidi, Steven Bethard, and Marie-Francine Moens. 2012. Semeval-2012 task 3: Spatial role labeling. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics*, pages 365–373, Montréal, Canada.

Egoitz Laparra and German Rigau. 2013. Sources of evidence for implicit argument resolution. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013) – Long Papers*, pages 155–166, Potsdam, Germany.

Bart Mellebeek, Francesc Benavent, Jens Grivolla, Joan Codina, Marta R Costa-Jussa, and Rafael Banchs. 2010. Opinion mining of spanish customer comments with non-expert annotations on mechanical turk. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*, pages 114–121, Los Angeles, CA.

Josef Ruppenhofer, Caroline Sporleder, R. Morante, Collin Baker, and Martha Palmer. 2010. Semeval-2010 task 10: Linking events and their participants in discourse. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 45–50, Uppsala, Sweden.

Carina Silberer and Anette Frank. 2012. Casting implicit role linking as an anaphora resolution task. In *Proceedings of SEM 2012: The First Joint Conference on Lexical and Computational Semantics*, pages 1–10, Montreal, Canada.

Coreference Resolution Evaluation for Higher Level Applications

Don Tuggener
Unversity of Zurich
Institute of Computational Linguistics
tuggener@cl.uzh.ch

Abstract

This paper presents an evaluation framework for coreference resolution geared towards interpretability for higher-level applications. Three application scenarios for coreference resolution are outlined and metrics for them are devised. The metrics provide detailed system analysis and aim at measuring the potential benefit of using coreference systems in preprocessing.

1 Introduction

Coreference Resolution is often described as an important preprocessing step for higher-level applications. However, the commonly used coreference evaluation metrics (MUC, BCUB, CEAF, BLANC) treat coreference as a generic clustering problem and perform cluster similarity measures to evaluate coreference system outputs. Mentions are seen as unsorted generic items rather than linearly ordered linguistic objects (Chen and Ng, 2013). This makes it arguably hard to interpret the scores and assess the potential benefit of using a coreference system as a preprocessing step.

Therefore, this paper proposes an evaluation framework for coreference systems which aims at bridging the gap between coreference system development, evaluation, and higher level applications. For this purpose, we outline three types of application scenarios which coreference resolution can benefit and devise metrics for them which are easy to interpret and provide detailed system output analysis based on any available mention feature.

2 Basic Concepts

Like other coreference metrics, we adapt the concepts of Recall and Precision from evaluation in Information Retrieval (IR) to compare mentions in a system output (the response) to the annotated mentions in a gold standard (the key). To stay close to the originally clear definitions of Recall and Precision in IR, Recall is aimed at identifying *how many of the annotated key mentions are correctly resolved* by a system, and Precision will measure *the correctness of the returned system mentions*.

However, if we define Recall as $\frac{tp}{tp+fn}$, the denominator will not include key mentions that have been put in the wrong coreference chain, and will not denote *all mentions in the key*. Therefore, borrowing nomenclature from (Durrett and Klein, 2013), we introduce an additional error class, *wrong linkage (wl)*, which signifies key mentions that have been linked to incorrect antecedents. Recall can then be defined as $\frac{tp}{tp+\mathbf{wl}+fn}$ and Precision as $\frac{tp}{tp+\mathbf{wl}+fp}$. Recall then extends over all key mentions, and Precision calculation includes all system mentions.

Furthermore, including *wrong linkage* in the Recall equation prevents it from inflating compared to Precision when a large number of key mentions are incorrectly resolved. Evaluation is also sensitive to the anaphoricity detection problem. For example, an incorrectly resolved *anaphoric* "it" pronoun is counted as *wrong linkage* and thus also affects Recall, while a resolved *pleonastic* "it" pronoun is considered a *false positive* which is only penalized by Precision. Beside the "it" pronoun, this is of particular relevance for noun markables, as determining their referential status is a non-trivial subtask in coreference resolution.

As we evaluate each mention individually, we are able to measure performance regarding any feature type of a mention, e.g. PoS, number, gender, semantic class etc. We will focus on mention types based on PoS tags (i.e. pronouns, nouns etc.), as they are often the building blocks of coreference systems. Furthermore, mention type based

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 231–235,
Gothenburg, Sweden, April 26-30 2014.

performance analysis is informative for higher-level applications, as they might be specifically interested in certain mention types.

3 Application Scenarios

Next, we will outline three higher-level application types which consume coreference and devise relevant metrics for them.

3.1 Models of entity distributions

The first application scenario subsumes models that investigate distributions and patterns of entity occurrences in discourse. For example, Centering theory (Grosz et al., 1995) and the thereof derived entity grid model (Barzilay and Lapata, 2008; Elsner and Charniak, 2011) record transitions of grammatical functions that entities occur with in coherent discourse. These models can benefit from coreference resolution if entities are pronominalized or occur as a non-string matching nominal mentions.

Another application which tracks sequences of entity occurrences is event sequence modeling. Such models investigate prototypical sequences of events to derive event schemes or templates of successive events (Lee et al., 2012; Irwin et al., 2011; Kuo and Chen, 2007). Here, coreference resolution can help link pronominalized arguments of events to their previous mention and, thereby, maintain the event argument sequence.

The outlined applications in this scenario primarily rely on the identification of *correct and gapless sequences of entity occurrences*. We can approximate this requirement in a metric by requiring the immediate antecedent of a mention in a response chain to be the immediate antecedent of that mention in the key chain.

Note that this restriction deems mentions as incorrect, if they skip an antecedent but are resolved to another antecedent in the correct chain. For example, given a key [A-B-C-D], mention D in a response [A-B-D] would not be considered correct, as the immediate antecedent is not the same as in the key. The original sequence of the entity's occurrence is broken between mention B and D in the response, as mention C is missing.

We use the following algorithm (table 1) to calculate Recall and Precision for evaluating immediate antecedents. Let K be the key and S be the system response. Let e be an entity denoted by m_n mentions.

01 for $e_k \in K$:
02 for $m_i \in e_k \wedge i > 0$:
03 if $\neg\exists e_s, m_j : (e_s \in S \wedge m_j \in e_s \wedge m_j = m_i \wedge \exists predecessor(m_j, e_s)) \rightarrow$ fn++
04 elif $\exists e_s, m_j : (e_s \in S \wedge m_j \in e_s \wedge m_j = m_i \wedge predecessor(m_i, e_k) = predecessor(m_j, e_s)) \rightarrow$ tp++
05 else wl++
06 for $e_s \in S$:
07 for $m_i \in e_s \wedge i > 0$:
08 if $\neg\exists e_k, m_j : (e_k \in K \wedge m_j \in e_k \wedge m_j = m_i \wedge \exists predecessor(m_j, e_k)) \rightarrow$ fp++

Table 1: Algorithm for calculating Recall and Precision.

We traverse the key K and each entity e_k in it[1]. We evaluate each mention m_i in e_k, except for the first one (line 2), as we investigate coreference links. If no response chain exists that contains m_i and its predecessor, we count m_i as a false negative (line 3). This condition subsumes the case where m_i is not in the response, and the case where m_i is the first mention of a response chain. In the latter case, the system has deemed m_i to be non-anaphoric (i.e. the starter of a chain), while it is anaphoric in the key[2]. We check whether the immediate predecessor of m_i in the key chain e_k is also the immediate predecessor of m_j in the response chain e_s (line 4). If true, we count m_i as a true positive, or as wrong linkage otherwise.

We traverse the response chains to detect spurious system mentions, i.e. mentions not in the key, and count them as false positives, i.e. non-anaphoric markables that have been resolved by the system (lines 6-8). Here, we also count mentions in the response, which have no predecessor in a key chain, as false positives. If a mention in the response chain is the chain starter in a key chain, it means that the system has falsely deemed it to be anaphoric and we regard it as a false positive[3].

3.2 Inferred local entities

The second application scenario relies on coreference resolution to infer local nominal antecedents. For example, in Summarization, a target sentence may contain a pronoun which should be replaced by a nominal antecedent to avoid ambiguities and ensure coherence in the summary. Machine Trans-

[1] We disregard singleton entities, as it is not clear what benefit a higher level application could gain from them.

[2] (Durrett and Klein, 2013) call this error *false new (FN)*.

[3] This error is called *false anaphoric (FA)* by (Durrett and Klein, 2013).

lation can benefit from pronoun resolution in language pairs where nouns have grammatical gender. In such language pairs, the gender of a pronoun antecedent has to be retrieved in the source language in order to insert the pronoun with the correct gender in the target language.

In these applications, it is not sufficient to link pronouns to other pronouns of the same coreference chain because they do not help infer the underlying entity. Therefore, in our metric, we require the closest preceding nominal antecedent of a mention in a response chain to be an antecedent in the key chain.

The algorithm for calculation of Recall and Precision is similar to the one in table 1. We modify lines 3 and 4 to require the closest nominal antecedent of m_i in the response chain e_s to be an antecedent of m_j in the corresponding key chain e_k, where $m_j = m_i$, i.e.:

$\exists m_h \in e_s : is\_closest\_noun(m_h, m_i) \wedge \exists e_k, m_j, m_l : (e_k \in K \wedge m_j \in e_k \wedge m_j = m_i \wedge m_l \in e_k \wedge l < j \wedge m_l = m_h) \rightarrow$ tp++

Note that we cannot process chains without a nominal mention in this scenario[4]. Therefore, we skip evaluation for such $e_k \in K$. We still want to find incorrectly inferred nominal antecedents of anaphoric mentions, i.e. mentions in $e_s \in S$ that have been assigned a nominal antecedent in the response but have none in the key and count them as wrong linkage, as they infer an incorrect nominal antecedent. Therefore, we traverse all $e_s \in S$ and add to the algorithm:

$\forall m_i \in e_s : \neg is\_noun(m_i) \wedge \exists m_h \in e_s : is\_noun(m_h) \wedge \exists e_k, m_j : (e_k \in K \wedge m_j \in e_k \wedge m_j = m_i \wedge \neg \exists m_l \in e_k : is\_noun(m_l)) \rightarrow$ wl++

3.3 Finding contexts for a specific entity

The last scenario we consider covers applications that are primarily query driven. Such applications search for references to a given entity and analyze or extract its occurrence contexts. For example, Sentiment Analysis searches large text collections for occurrences of a target entity and then derives polarity information from its contexts. Biomedical relation mining looks for interaction contexts of specific genes or proteins etc.

[4]We found that 476 of 4532 key chains (10.05%) do not contain a nominal mention. Furthermore, we do not treat cataphora (i.e. pronouns at chain start) in this scenario. We found that 241 (5.31%) of the key chains start with cataphoric pronouns.

For these applications, references to relevant entities have to be accessible by queries. For example, if a sentiment system investigates polarity contexts of the entity "Barack Obama", given a key chain [Obama - the president - he], a response chain [the president - he] is not sufficient, because the higher level application is not looking for instances of the generic "president" entity.

Therefore, we determine an *anchor mention* for each coreference chain which represents the most likely unique surface form an entity occurs with. As a simple approximation, we choose the first nominal mention of a coreference chain to be the anchor of the entity, because first mentions of entities introduce them to discourse and are, therefore, generally informative, unambiguous, semantically extensive and are likely to contain surface forms a higher level application will query.

Entity Detection
01 for $e_k \in K$:
02 if $\exists m_n \in e_k : is\_noun(m_n)$ $\rightarrow m_{anchor} = determine\_anchor(e_k)$
03 if $\exists m_{anchor} \wedge \exists e_s \in S : m_{anchor} \in e_s \rightarrow$ tp++
04 else $\rightarrow$ fn++
05 for $e_s \in S$:
06 if $\exists m_n \in e_s : is\_noun(m_n)$ $\rightarrow m_{anchor} = determine\_anchor(e_s)$
07 if $\neg \exists e_k \in K : m_{anchor} \in e_k \rightarrow$ fp++
Entity Mentions
01 for $e_k \in K : \exists m_{anchor} \wedge \exists e_s \in S : m_{anchor} \in e_s :$
02 for $m_i \in e_k$:
03 if $m_i \in e_s \rightarrow$ tp++
04 else$\rightarrow$ fn++
05 for $m_i \in e_s$:
06 if $m_i \neg \in e_k \rightarrow$ fp++

Table 2: Algorithm for calculating Recall and Precision using anchor mentions.

To calculate Recall and Precision, we align coreference chains in the responses to those in the key via their anchors and then measure how many (in)correct references to that anchor the coreference systems find (table 2). We divide evaluation into *entity detection* (ED), which measures how many of the anchor mentions a system identifies. We then measure the quality of the *entity mentions* (EM) for only those entities which have been aligned through their anchors.

The quality of the references to the anchor mentions are not directly comparable between systems, as their basis is not the same if the number of aligned anchors differs. Therefore, we calculate the harmonic mean of entity detection and entity mentions to enable direct system compari-

son. Where applicable, we obtain the named entity class of the entity and measure performance for each such class.

4 Evaluation

We apply our metrics to three available coreference systems, namely the Berkley system (Durrett and Klein, 2013), the IMS system (Björkelund and Farkas, 2012), and the Stanford system (Lee et al., 2013) and their responses for the CoNLL 2012 shared task test set for English (Pradhan et al., 2012). Tables 3 and 4 report the results.

	Immediate antecedent			Inferred antecedent		
	R	P	F	R	P	F
	BERK (Durrett and Klein, 2013)					
NOUN	45.06	47.06	46.04	55.54	60.37	57.85
PRP	67.66	64.87	66.24	48.92	53.62	51.16
PRP$	74.49	74.32	74.41	61.95	66.80	64.28
TOTAL	56.60	56.91	56.76	52.94	58.04	55.37
	IMS (Björkelund and Farkas, 2012)					
NOUN	38.01	43.09	40.39	46.90	54.96	50.61
PRP	**69.06**	**68.64**	**68.85**	43.04	**57.42**	49.20
PRP$	72.57	72.11	72.34	51.51	63.54	56.90
TOTAL	53.55	57.55	55.48	45.27	56.47	50.25
	STAN (Lee et al., 2013)					
NOUN	38.51	42.92	40.60	50.03	57.62	53.56
PRP	65.55	61.09	63.25	36.67	45.97	40.80
PRP$	66.12	65.70	65.91	40.64	52.38	45.77
TOTAL	51.70	52.69	52.19	43.01	51.73	46.97

Table 3: Antecedent based evaluation

We note that the system ranking based on the MELA score[5] is retained by our metrics. MELA rates the Berkley system best (61.62), followed by the IMS system (57.42), and then the Stanford system (55.69).

Beside detailed analysis based on PoS tags, our metrics reveal interesting nuances. Somewhat expectedly, noun resolution is worse when the immediate antecedent is evaluated, than if the next nominal antecedent is analyzed. Symmetrically inverse, pronouns achieve higher scores when their direct antecedent is measured, as compared to when the next nominal antecedent has to be correct.

Our evaluation shows that the IMS system achieves a higher score for pronouns than the Berkley system when immediate antecedents are measured and has a higher Precision for pronouns regarding the inferred antecedents. The Berkley system performs best mainly due to Recall. For e.g. personal pronouns (PRP), Berkley has the following counts for the inferred antecedents: tp=2687, wl=1935, **fn=871**, fp=389, while IMS shows tp=2243, wl=1376, **fn=1592**, fp=287. This indicates that the IMS Recall is lower because of the high false negative count, rather than being due to too many wrong linkages.

Finally, table 4 suggests that the IMS systems performs significantly worse in the PERSON class than the other systems and is outperformed by the Stanford system in the ORG class, but performs best in the GPE class.

		R	P	F	Fϕ
		PERSON (18.69%)			
BERK	ED	64.02	75.88	69.45	67.11
	EM	63.60	66.29	64.92	
IMS	ED	45.66	51.69	48.48	**52.74**
	EM	47.67	73.45	57.82	
STAN	ED	56.33	59.74	57.98	61.61
	EM	53.84	84.37	65.73	
		GPE (13.28%)			
BERK	ED	73.21	77.36	75.23	75.71
	EM	69.89	83.73	76.19	
IMS	ED	73.51	74.17	73.84	**76.21**
	EM	69.94	90.04	78.73	
STAN	ED	70.24	76.62	73.29	75.24
	EM	68.44	88.81	77.30	
		ORG (9.63%)			
BERK	ED	62.78	67.13	64.88	67.62
	EM	66.87	74.78	70.60	
IMS	ED	44.98	54.30	49.20	56.85
	EM	57.26	81.66	67.32	
STAN	ED	49.68	58.56	53.75	**59.41**
	EM	57.25	79.05	66.41	
		TOTAL (100%)			
BERK	ED	58.65	53.19	55.79	63.41
	EM	72.65	74.28	73.45	
IMS	ED	47.16	42.66	44.80	55.24
	EM	65.88	79.40	72.01	
STAN	ED	48.62	41.40	44.72	55.27
	EM	65.66	80.48	72.32	

Table 4: Anchor mention based evaluation

5 Conclusion

We have presented a simple evaluation framework for coreference evaluation with higher level applications in mind. The metrics allow specific performance measurement regarding different antecedent requirements and any mention feature, such as PoS type, lemma, or named entity class, which can aid system development and comparison. Furthermore, the metrics do not alter system rankings compared to the commonly used evaluation approach[6].

[5] $\frac{MUC+BCUB+CEAFE}{3}$

[6] The scorers are freely available on our website: http://www.cl.uzh.ch/research/coreferenceresolution.html

References

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Comput. Linguist.*, 34(1):1–34, March.

Anders Björkelund and Richárd Farkas. 2012. Data-driven multilingual coreference resolution using resolver stacking. In *Joint Conference on EMNLP and CoNLL - Shared Task*, CoNLL '12, pages 49–55, Stroudsburg, PA, USA. Association for Computational Linguistics.

Chen Chen and Vincent Ng. 2013. Linguistically aware coreference evaluation metrics. In *Proceedings of the 6th International Joint Conference on Natural Language Processing*, pages 1366–1374.

Greg Durrett and Dan Klein. 2013. Easy victories and uphill battles in coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, October. Association for Computational Linguistics.

Micha Elsner and Eugene Charniak. 2011. Extending the entity grid with entity-specific features. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 125–129, Stroudsburg, PA, USA. Association for Computational Linguistics.

Barbara J. Grosz, Scott Weinstein, and Aravind K. Joshi. 1995. Centering: a framework for modeling the local coherence of discourse. *Comput. Linguist.*, 21(2):203–225, June.

Joseph Irwin, Mamoru Komachi, and Yuji Matsumoto. 2011. Narrative schema as world knowledge for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, CONLL Shared Task '11, pages 86–92, Stroudsburg, PA, USA. Association for Computational Linguistics.

June-Jei Kuo and Hsin-Hsi Chen. 2007. Cross-document event clustering using knowledge mining from co-reference chains. *Inf. Process. Manage.*, 43(2):327–343, March.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 489–500, Stroudsburg, PA, USA. Association for Computational Linguistics.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics*, 39(4).

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes. In *Proceedings of the Sixteenth Conference on Computational Natural Language Learning (CoNLL 2012)*, Jeju, Korea.

Efficient Online Summarization of Microblogging Streams

Andrei Olariu
Faculty of Mathematics and Computer Science
University of Bucharest
andrei@olariu.org

Abstract

The large amounts of data generated on microblogging services are making summarization challenging. Previous research has mostly focused on working in batches or with filtered streams. Input data has to be saved and analyzed several times, in order to detect underlying events and then summarize them. We improve the efficiency of this process by designing an online abstractive algorithm. Processing is done in a single pass, removing the need to save any input data and improving the running time. An online approach is also able to generate the summaries in real time, using the latest information. The algorithm we propose uses a word graph, along with optimization techniques such as decaying windows and pruning. It outperforms the baseline in terms of summary quality, as well as time and memory efficiency.

1 Introduction

Coined in 2006-2007, the term microblogging is used to describe social networks that allow users to exchange small elements of content. The widespread use of services like Facebook or Twitter means users have access to information that is otherwise unavailable. Yet, as popular events commonly generate hundreds of thousands of tweets, following them can be difficult. Stream summarization – generating a short text based on a sequence of posts – has been seen as the best approach in solving this problem.

This paper introduces Twitter Online Word Graph Summarizer. TOWGS is the first online abstractive summarization algorithm and is capable of state-of-the-art processing speeds. Most previous algorithms process a stream in batches. They require several passes through the data or a feed specifically filtered for an event. Batch summarization is suitable for small experiments, but it is not capable of efficiently handling thousands of tweets per second.

We collect a 3.4 million tweets dataset for evaluation purposes. We choose as baseline an algorithm designed to summarize related tweets. We determine a set of important events relative to the input data. A group of judges rate the summaries generated by both algorithms for the given events. Our solution is not only capable of online summarization, but it also outperforms the batch-based event-filtered baseline in terms of result quality. The code for our algorithm is available online, along with the summaries, event keywords, ratings and tweet IDs: `https://github.com/andreiolariu/online-summarizer`.

2 Related Work

2.1 Summarization

We distinguish two approaches in performing multi-document summarization: extractive and abstractive. With the risk of oversimplification, we view extractive summarization as a process of selecting sentences from the documents, while abstractive summarization generates phrases that may not appear in the input data.

The extractive approach is usually modeled as an optimization problem (Erkan and Radev, 2004). It can be combined with other techniques, such as clustering (Silveira and Branco, 2012) or topic modeling (Li and Li, 2013).

Although actually performing word-level extraction, we consider word graph summarization algorithms abstractive because they are able to generate summaries not found among the input sentences. Word graphs are used in compressing similar sentences (Filippova, 2010) or summarizing product reviews (Ganesan et al., 2010).

A relevant reference for the problem of up-

Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, pages 236–240,
Gothenburg, Sweden, April 26-30 2014.

date summarization is TAC update summarization track (Dang and Owczarzak, 2008).

2.2 Summarization on Twitter

Regarding summarizing Twitter streams, we notice that all approaches are either restricted to specific filtered streams, or combined with event detection.

Extractive summarization is predominant when working with Twitter data. It was first used for streams following simple and structured events, such as sports matches (Takamura et al., 2011; Chakrabarti and Punera, 2011; Nichols et al., 2012; Zubiaga et al., 2012).

The Phrase Reinforcement algorithm, introduced by Sharifi et al. (2010a; 2010b), extracts frequently used sequences of words. It was first applied in summarizing topic streams. Subsequent research emphasized evolving topics (Gao et al., 2013) or event decomposition (Olariu, 2013).

Other approaches are based on integer linear programming (Liu et al., 2011) or LDA (Khan et al., 2013). Yang et al. (2012) develop a framework for summarization, highlighting its scalability. Shou et al. (2013) introduce Sumblr, capable of cluster-based online extractive summarization.

Abstractive summarization is difficult on Twitter streams. It is easily affected by noise or by the large variety of tweets. Olariu (2013) showed that abstractive summarization is feasible if posts are clustered based on similarity or underlying events.

3 Twitter Online Word Graph Summarizer

3.1 Building the Word Graph

By employing a word graph, TOWGS doesn't have to save any of the tweets, like extractive approaches do. It can also skip the clustering step applied by the other online algorithm (Shou et al., 2013), leading to faster summarization.

Previous word graph algorithms are based on bigrams. Words are mapped to nodes in the graph, while an edge is added for each bigram. When applied to Twitter messages, the results depend on the similarity of the summarized tweets (Olariu, 2013). A set of related tweets generates a quality summary. When applied to unrelated tweets, the generated summary lacks any meaning. This happens because event-related signals (in our case bigrams) stand out when analyzing similar tweets, but get dominated by noise (bigrams of common words) when analyzing unrelated tweets.

We solve this issue by building the word graph from trigrams. In our version, each node in the graph is a bigram. Having a sentence (w_1, w_2, w_3, w_4), we will first add two special words (to mark the beginning and end of the sentence) and generate the following edges: $(S, w_1) \rightarrow (w_1, w_2)$, $(w_1, w_2) \rightarrow (w_2, w_3)$, $(w_2, w_3) \rightarrow (w_3, w_4)$ and $(w_3, w_4) \rightarrow (w_4, E)$. Weights are added to nodes and edges in order to store the count for each bigram or trigram.

A negative effect of building the word graph from trigrams is that it significantly increases the number of nodes, leading to an increase in both memory and time. We approach this issue by pruning the graph. We implement pruning by periodically going through the whole graph and removing edges that were not encountered in the previous time window. The length of this hard window can be set based on how much memory we would like to allocate, as well as on the size of the soft window introduced in the next subsection.

3.2 Word Graph Online Updating

In previous work, word graphs are discarded after generating the summary. For our online summarization task, the graph is being constantly updated with tweets. It can also respond, at any time, to queries for generating summaries starting from given keywords.

In order to keep the results relevant to what is popular at query time, we would like the graph to *forget* old data. We implement this behavior by using decaying windows (Rajaraman and Ullman, 2011). They are applied not only to graph weights (counts of bigrams and trigrams), but also to counts of words and word pair cooccurrences.

At each time step (in our case, each second), all counts are multiplied by $1 - c$, where c is a small constant. For example, after one hour (3600 seconds), a value of 1 would become 0.48 with $c = 0.0002$ (given by $(1 - c)^{3600}$) and 0.05 with $c = 0.0008$.

In order to optimize the implementation, we explicitly multiply the counts only when they are read or incremented. For each record, we keep the timestamp for its latest update t_k. Knowing the current timestamp t_n, we update the count by multiplying with $(1 - c)^{t_n - t_k}$.

The size of the decaying window influences the

results and the memory requirements for TOWGS. A larger window requires less pruning and more memory, while also leading to more general summaries. For example, given a stream of tweets related to a sporting event, summaries generated over very narrow windows would probably highlight individual goals, touchdowns or penalties. The summary for a two hour window would instead capture just the final score.

3.3 Generating Summaries

Given a word graph, generating a summary involves finding the highest scoring path in the graph. That path connects the special words which mark the beginning and end of each sentence. Since finding the exact solution is unfeasible given our real time querying scenario, we will employ a greedy search strategy.

The search starts by selecting the node (bigram) with the highest weight. If we are interested in summarizing an event, we select the top ranking bigram containing one of the event's keywords.

At this point, we have a path with one node. We expand it by examining forward and backward edges and selecting the one that maximizes the scoring function:

$$
\begin{aligned}
&score(n, e, m, p, k) = \\
&\quad c_1\, frequency(n) && \text{(1a)}\\
&\quad + c_2\, edge\_score(e, m) && \text{(1b)}\\
&\quad + c_3\, word\_score(n, p) && \text{(1c)}\\
&\quad + c_4\, word\_score(n, k) && \text{(1d)}\\
&\quad - c_5\, frequent\_word\_pen(n) && \text{(1e)}\\
&\quad - c_6\, repeated\_word\_pen(n) && \text{(1f)}
\end{aligned}
$$

where p is a path representing a partial summary, n is a node adjacent to one of the path's endpoints m by edge e and k is a list of keywords related to an event. The constants c_1 through c_6 determine the influence each helper function has on the overall score. The node n represents a bigram composed of the words w_i (already in the path as part of m) and w_o (currently being considered for extending p). The helper functions are defined as:

$$frequency(n) = \log(W_b[n]) \quad \text{(2a)}$$

$$edge\_score(e, m) = \log\left(\frac{W_t[e]}{W_b[m]}\right) \quad \text{(2b)}$$

$$word\_score(n, p) = \sum_{w \in p} \frac{1}{|p|} \log\left(\frac{W_d[w, w_o]}{\sqrt{W_w[w] W_w[w_o]}}\right) \quad \text{(2c)}$$

$$frequent\_word\_pen(n) = \log(W_w[w_o]) \quad \text{(2d)}$$

$$repeated\_word\_pen(n) = \mathbf{1}_p(w_o) \quad \text{(2e)}$$

where $W_w[w]$ is the weight for word w, $W_b[m]$ is the weight for the bigram represented by node m, $W_t[e]$ is the weight for the trigram represented by edge e and $W_d[w, w_o]$ is the weight for the cooccurrences of words w and w_o in the same tweets. $\mathbf{1}_p(w_o)$ is the indicator function. In all these cases, weights are counts implemented using decaying windows (subsection 3.2).

The scoring function gives a higher score to frequent bigrams (equations 1a and 2a). In the same time, individual words are penalized on their frequency (equations 1e and 2d). Such scores favor words used in specific contexts as opposed to general ones. Trigrams are scored relative to bigrams (equations 1b and 2b). Again, this favors context specific bigrams. The word score function (equation 2c) computes the average correlation between a word (w_o from the bigram represented by node n) and a set of words. The set of words is either the current partial summary (equation 1c) or the event-related keywords (equation 1d).

We use logarithms in order to avoid floating point precision errors.

4 Evaluation

4.1 Corpus and Baseline

Our corpus is built using the Twitter Search API. We gathered an average of 485000 tweets per day for a total of seven days, between the 4th and the 10th of November 2013. This volume of tweets represents around 0.1% of the entire Twitter stream. Because of Twitter's terms of service, sharing tweets directly is not allowed. Instead, the source code we've released comes with the tweet IDs needed for rebuilding the corpus.

The algorithm chosen as baseline is Multi-Sentence Compression (or MSC), as presented in (Olariu, 2013). MSC is a batch algorithm for abstractive summarization. It performs best on groups of similar tweets, such as the ones related to an event. After receiving a summarization query for a set of keywords, the tweets are filtered based on those keywords. MSC processes the remaining tweets and generates a word graph. After building the summary, the graph is discarded.

Because it has to store all tweets, MSC is not as memory-efficient as TOWGS. It is also not time-efficient. Each summarization query requires fil-

tering the whole stream and building a new word graph. The advantage MSC has is that it is working with filtered data. Olariu (2013) has shown how susceptible word graphs are to noise.

4.2 Evaluation Procedure

The list of 64 events to be summarized was determined using a frequency based approach. A simple procedure identified words that were used significantly more in a given day compared to a baseline. The baselines were computed on a set of tweets posted between the 1st and the 3rd of November 2013. Words that often appeared together were grouped, with each group representing a different event.

The MSC algorithm received a cluster of posts for each event and generated summaries of one sentence each. TOWGS processed the posts as a stream and answered to summarization requests. The requests were sent after the peak of each event (at the end of the hour during which that event registered the largest volume of posts).

The metrics used for assessing summary quality were completeness (how much information is expressed in the summary, relative to the event tweets) and grammaticality. They were rated on a scale of 1 (lowest) to 5 (highest).

We asked five judges to rate the summaries using a custom built web interface. The judges were not native English speakers, but they were all proficient. Three of them were Twitter users. While the judges were subjective in assessing summary quality, each one did rate all of the summaries and the differences between the two algorithms' ratings were consistent across all judges.

The constants c_1 through c_6 (introduced in subsection 3.3) were set to 2, 3, 3, 10, 1 and 100, respectively. These values were manually determined after experimenting with a one day sample not included in the evaluation corpus.

5 Results

The average ratings for completeness are very similar, with a small advantage for TOWGS (4.29 versus MSC's 4.16). We believe this is a good result, considering TOWGS doesn't perform clustering and summarizes events that account for less than 1% of the total volume. Meanwhile, MSC processes only the event-related tweets. The average rating for grammaticality is significantly higher for TOWGS (4.30), as compared to MSC (3.78).

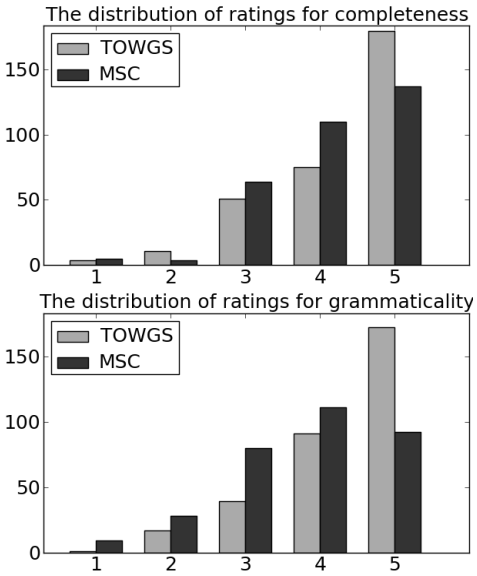


Figure 1: The ratings distribution by algorithm and metric.

While not engineered for speed, our implementation can process a day of data from our corpus (around 485000 tweets) in just under three minutes (using one 3.2 GHz core). In comparison, Sumblr (Shou et al., 2013) can process around 30000 tweets during the same interval. TOWGS requires an average of 0.5 seconds for answering each summarization query. Regarding memory use, pruning kept its value constant. In our experiments, the amount of RAM used by the algorithm was between 1.5 - 2 GB.

The code for TOWGS is available online, along with the summaries, keywords, ratings and tweet IDs: `https://github.com/andreiolariu/online-summarizer`.

6 Conclusion

Summarizing tweets has been a popular research topic in the past three years. Yet developing efficient algorithms has proven a challenge, with most work focused on small filtered streams.

This paper introduces TOWGS, a highly efficient algorithm capable of online abstractive microblog summarization. TOWGS was tested on a seven day 0.1% sample of the entire Twitter stream. We asked five judges to rate the summaries it generated, along with those from a baseline algorithm (MSC). After aggregating the results, the summaries generated by TOWGS proved to have a higher quality, despite the fact that MSC processed just the batches of event-filtered tweets. We also highlighted the state-of-the-art time efficiency of our approach.

References

Deepayan Chakrabarti and Kunal Punera. 2011. Event summarization using tweets. In *Proceedings of the 5th Int'l AAAI Conference on Weblogs and Social Media (ICWSM)*.

Hoa Trang Dang and Karolina Owczarzak. 2008. Overview of the tac 2008 update summarization task. In *Proceedings of text analysis conference*, pages 1–16.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.*, 22(1):457–479, December.

Katja Filippova. 2010. Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 322–330, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 340–348, Stroudsburg, PA, USA. Association for Computational Linguistics.

Dehong Gao, Wenjie Li, and Renxian Zhang. 2013. Sequential summarization: A new application for timely updated twitter trending topics. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 567–571. Association for Computational Linguistics.

Muhammad Asif Hossain Khan, Danushka Bollegala, Guangwen Liu, and Kaoru Sezaki. 2013. Multi-tweet summarization of real-time events. In *SocialCom*, pages 128–133. IEEE.

Jiwei Li and Sujian Li. 2013. Evolutionary hierarchical dirichlet process for timeline summarization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 556–560. Association for Computational Linguistics.

Fei Liu, Yang Liu, and Fuliang Weng. 2011. Why is "sxsw" trending?: exploring multiple text sources for twitter topic summarization. In *Proceedings of the Workshop on Languages in Social Media*, LSM '11, pages 66–75, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jeffrey Nichols, Jalal Mahmud, and Clemens Drews. 2012. Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, IUI '12, pages 189–198, New York, NY, USA. ACM.

Andrei Olariu. 2013. Hierarchical clustering in improving microblog stream summarization. In *Proceedings of the 14th international conference on Computational Linguistics and Intelligent Text Processing - Volume 2*, CICLing'13, pages 424–435, Berlin, Heidelberg. Springer-Verlag.

Anand Rajaraman and Jeffrey David Ullman. 2011. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA.

Beaux Sharifi, Mark-Anthony Hutton, and Jugal Kalita. 2010a. Summarizing microblogs automatically. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 685–688, Stroudsburg, PA, USA. Association for Computational Linguistics.

Beaux Sharifi, Mark-Anthony Hutton, and Jugal K. Kalita. 2010b. Experiments in microblog summarization. In *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, SOCIALCOM '10, pages 49–56, Washington, DC, USA. IEEE Computer Society.

Lidan Shou, Zhenhua Wang, Ke Chen, and Gang Chen. 2013. Sumblr: Continuous summarization of evolving tweet streams. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '13, pages 533–542, New York, NY, USA. ACM.

S.B. Silveira and A. Branco. 2012. Combining a double clustering approach with sentence simplification to produce highly informative multi-document summaries. In *Information Reuse and Integration (IRI), 2012 IEEE 13th International Conference on*, pages 482–489.

Hiroya Takamura, Hikaru Yokono, and Manabu Okumura. 2011. Summarizing a document stream. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 177–188, Berlin, Heidelberg. Springer-Verlag.

Xintian Yang, Amol Ghoting, Yiye Ruan, and Srinivasan Parthasarathy. 2012. A framework for summarizing and analyzing twitter feeds. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '12, pages 370–378, New York, NY, USA. ACM.

Arkaitz Zubiaga, Damiano Spina, Enrique Amigó, and Julio Gonzalo. 2012. Towards real-time summarization of scheduled events from twitter streams. In *Proceedings of the 23rd ACM Conference on Hypertext and Social Media*, HT '12, pages 319–320, New York, NY, USA. ACM.

Author Index

Aletras, Nikolaos, 22
Alexopoulos, Panos, 33
Ambati, Bharat Ram, 159
Ananiadou, Sophia, 111
Ananthakrishnan, Sankaranarayanan, 54
Asghari, Habibollah, 138

Baldwin, Timothy, 215
Bontcheva, Kalina, 69
Brandes, Jasper, 117

Casacuberta, Francisco, 90
Charniak, Eugene, 169
Chen, Hsin-Hsi, 12
Cho, Eunah, 43
Ciobanu, Alina Maria, 17, 64
Collins, Michael, 180
Cook, Paul, 215

Dara, Aswarth Abhilash, 185
Deoskar, Tejaswini, 159
Derczynski, Leon, 69
Dias, Gaël, 1, 6
Dinu, Anca, 64
Dinu, Liviu, 17, 64
Dras, Mark, 95
Duh, Kevin, 154, 190
Dunietz, Jesse, 205
Durrani, Nadir, 148

Esparza, Javier, 164
Etzioni, Oren, 12

Fader, Anthony, 12
Faili, Heshaam, 138
Feizabadi, Parvin Sadat, 226
Ferrari, Stéphane, 6

Gella, Spandana, 215
Gertz, Michael, 133
Gesmundo, Andrea, 28
Gillick, Daniel, 205
Gkatzia, Dimitra, 210
González-Rubio, Jesús, 90

Hall, Keith, 28
Hasanuzzaman, Mohammed, 6
Hastie, Helen, 210
Hewavitharana, Sanjika, 54
Hoang, Hieu, 148
Hwang, Seung-won, 59

Illig, Jens, 100

Judge, John, 185

Klakow, Dietrich, 100
Koehn, Philipp, 148
Kohonen, Oskar, 84
Kontonatsios, Georgios, 111
Kopeć, Mateusz, 221
Korkontzelos, Ioannis, 111
kurimo, mikko, 74, 84

Lee, Lung-Hao, 12
Lee, Sunyou, 59
Lee, Taesung, 59
Lemon, Oliver, 210
Lenci, Alessandro, 38
Li, Hui, 133
Liao, Bo-Shun, 12
Lin, Shu-Yen, 12
Linden, Krister, 74
Lipenkova, Janna, 143
Liu, Mei-Jun, 12
Liu, Qun, 185
Lu, Qin, 38
Luttenberger, Michael, 164

Makino, Takuya, 106
Maleki, Jalal, 138
Malmasi, Shervin, 95
Mathet, Yann, 6
Matsumoto, Yuji, 154, 190
Mehay, Dennis, 54
Moreno, Jose G., 1

Nakamura, Satoshi, 128
Neubig, Graham, 128
Ney, Hermann, 174
Niculae, Vlad, 17

Niehues, Jan, 43

Olariu, Andrei, 236
Östling, Robert, 123
Ouchi, Hiroki, 154

Padó, Sebastian, 226
Pavlopoulos, John, 33
Peitz, Stephan, 174

Rasooli, Mohammad Sadegh, 48
Roth, Benjamin, 100
Ruokolainen, Teemu, 74, 84
Ruppenhofer, Josef, 117

Sajjad, Hassan, 148
Sakti, Sakriani, 128
Santus, Enrico, 38
Sassano, Manabu, 79
Scheible, Christian, 200
Schlund, Maximilian, 164
Schulte im Walde, Sabine, 38
Schütze, Hinrich, 200
Silfverberg, Miikka, 74
Simion, Andrei, 180
Souček, Milan, 143
Specht, Günther, 195
Steedman, Mark, 159
Stein, Cliff, 180
Stevenson, Mark, 22
Strötgen, Jannik, 133
Swanson, Ben, 169

Tetreault, Joel, 48
Toda, Tomoki, 128
Toral, Antonio, 185
Tschuggnall, Michael, 195
Tseng, Yuen-Hsien, 12
Tsujii, Jun'ichi, 111
Tuggener, Don, 231

van Genabith, Josef, 185
Vilar, David, 174
Virpioja, Sami, 84
Vu, Hoa Trong, 128

Waibel, Alex, 43
Wiegand, Michael, 117

Yung, Frances, 190

Zampieri, Marcos, 17
Zell, Julian, 133